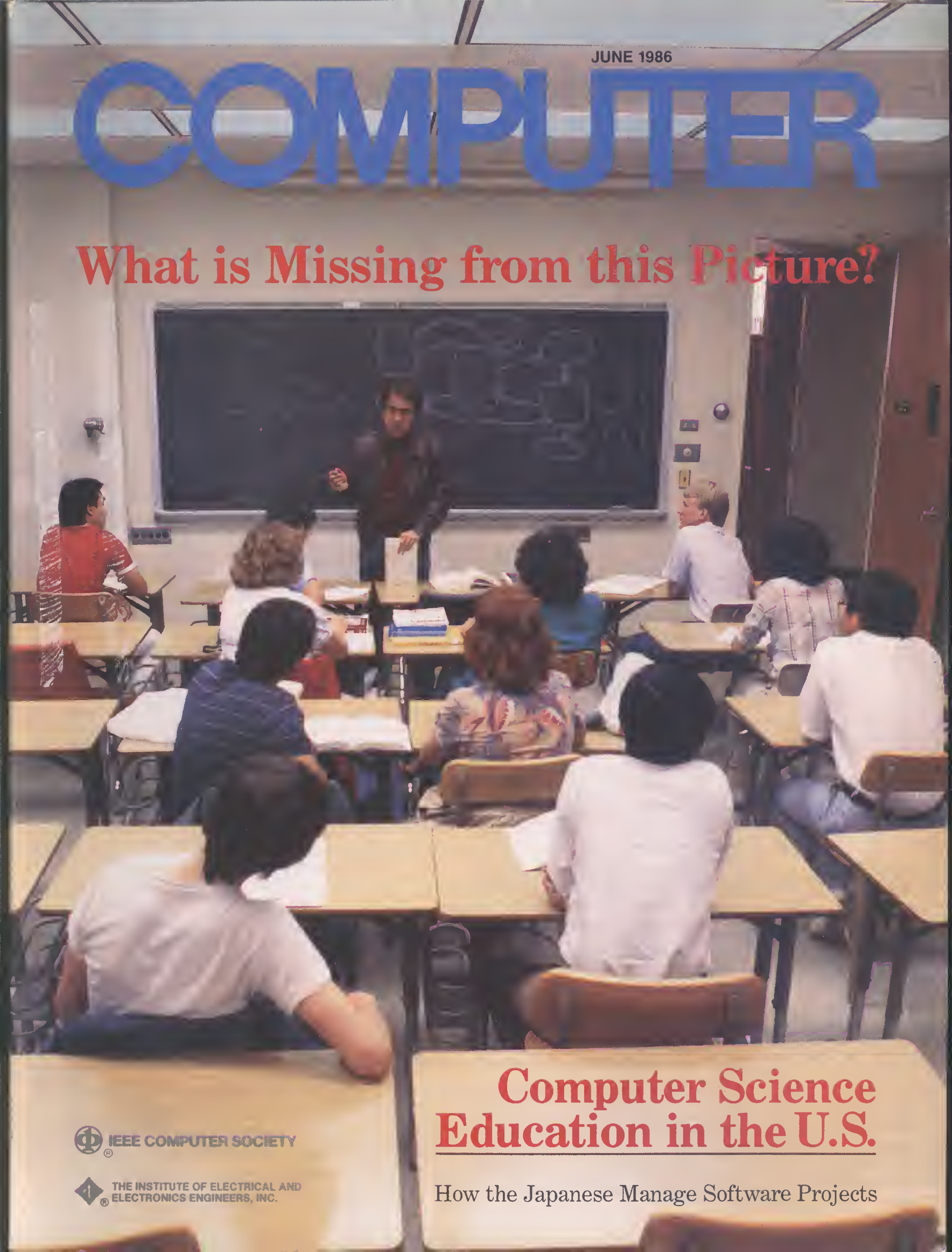
JUNE 1986

COMPUTER

What is Missing from this Picture?



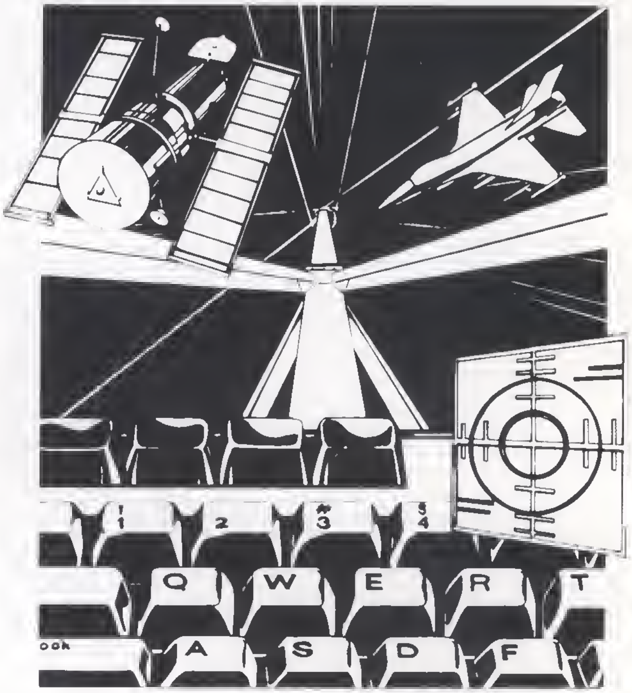
Computer Science Education in the U.S.

How the Japanese Manage Software Projects

IEEE COMPUTER SOCIETY

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

Computer Professionals



BDM wants to talk with you if you're looking for a fast-paced, innovative company. We invite you to use your academic background and entrepreneurial spirit to solve the problems that our clients bring to us. We are committed to strengthening the nation in meaningful ways—in defense, communications, space and many other areas. We take our capabilities and services directly to our clients. By using state-of-the-art technologies, we are able to find new, more effective solutions to difficult problems.

A broad spectrum of technologically challenging opportunities await you at our 36 nationwide locations. We offer you your choice of locations including McLean, VA; Albuquerque, NM; Los Angeles, CA; Huntsville, AL; Dayton, OH; Columbia, MD; Seattle, Ft. Lewis, and Tacoma, WA; and Austin, TX. Positions are available in the following areas:

Artificial Intelligence Engineers
- Expert Systems
- Advanced Computer Architecture
- Knowledge Representation & Acquisition
- LISP, KEE, Symbolics 3600s
- Military Systems/Battle Management Applications

Software Engineers
- Real-Time Software Development
- LANs
- TCP/IP, Ethernet, DECNET
- System Test and Integration
- Intelligence Fusion
- Configuration Management
- Quality Assurance
- Computer Security
- CAD/CAM
- Hardware/Software Integration
- Computer Integrated Manufacturing

Data Base Management Systems Leaders
- Micro-Based Data Management Systems
- FORTRAN, COBOL, Pascal, "C"
- VAX/VMS/MVS/UNIX
- Large Data Base Management Systems
- DATACOM/DB
- PSL/PSA
- Yourdon/DeMarco Structured Methodologies
- ORACLE
- Model 204
- Relational Data Base Management Systems

These are prime opportunities for Senior Engineers, Computer Scientists, and Information Systems professionals, preferably with an MS or PhD and 3 or more years of experience. Please send your resume with location preference to:

The BDM Corporation
Holly C. Marcario
7915 Jones Branch Dr.
McLean, VA 22102

The BDM Corporation
Judy C. Salmon
10116-36th Ave., CTSW
Tacoma, WA 98499

The BDM Corporation
Deanna P. Weaver
Dept. CM-0686
2227 Drake Ave.
Huntsville, AL 35805

The BDM Corporation
Ann C. Greenberg
1801 Randolph Rd., SE
Albuquerque, NM 87106

The BDM Corporation
Mary C. Eckhardt
6104 Poe Ave.
Dayton, OH 45414

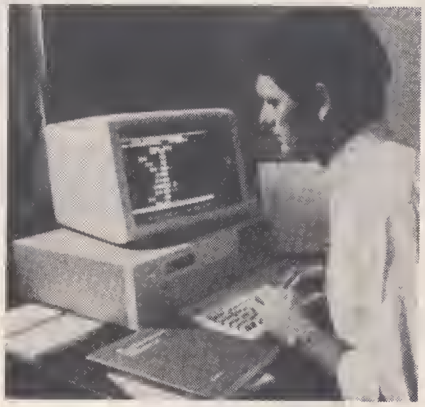
The BDM Corporation
Lilly C. Hill
5155 West Rosecrans Ave.
Hawthorne, CA 90250

Equal Opportunity Employer. U.S. Citizenship required. Subsidiary of BDM International, Inc.

BDM

Your Future Is Too Important To Spend Anywhere Else.

Announcing new PC SIMSCRIPT II.5 . . . with animation



SIMSCRIPT II.5 with animation now on personal computers

free trial--see how SIMSCRIPT II.5 helps you build a realistic model

the complete SIMSCRIPT II.5 on a PC

SIMSCRIPT II.5 for personal computers is the same popular simulation language that is now widely used on mainframes.

You can now build realistic models of military, manufacturing, communications, logistics, transportation or other systems on your PC.

PC SIMSCRIPT includes a new programming environment that makes it easy for you to develop, verify, modify, and enhance simulation models on a personal computer.

natural method of modelling

When building a model in SIMSCRIPT, you describe the simulated system as consisting of certain types of entities: perhaps workers, machines and jobs in a simulated factory; or flights and airports in a simulated air transport system; or jobs, processors, channels, and I/O devices in a simulated computer system.

For each type of entity you give names to the attributes that characterize it.

You also name the sets an entity type may belong to, and the sets it may own.

Since your model is English-like, with names that you choose, it reads like a description of the simulated system. The model can be read and verified by non-programmers who understand the system under study.

This makes your model development, validation and evolutionary changes much easier.

large models on your PC

Your model and data are not limited by the size of the PC. SIMSCRIPT is the only simulation tool that automatically makes use of the hard disk as a memory extension.

reduced cost

SIMSCRIPT II.5® is a well established, standardized, and widely used language with proven software support.

Experience has shown that SIMSCRIPT II.5 reduces simulation programming time and cost severalfold compared to other simulation techniques.

animated and graphical results

With PC SIMSCRIPT II.5® you build models that can show an animated picture of the system under study. Observing the simulation improves understanding of the system and builds confidence in the model.

Because you see the operation of the simulated system and can easily try alternatives, the time and cost of system analysis are sharply reduced.

computers with SIMSCRIPT II.5

1. IBM Personal Computer AT, XT, PC or compatible, with a hard disk.
2. Most Mainframe computer types including IBM, CDC, VAX, Univac, Prime, Gould, Data General and Honeywell.

SIMSCRIPT II.5 and PC SIMSCRIPT II.5 are registered trademarks and service marks of CACI, INC.-FEDERAL

free trial

The free trial package contains everything you need to try SIMSCRIPT II.5 on your own computer.

We send you PC or Mainframe SIMSCRIPT II.5, installation instructions, sample models, and a complete set of documentation. You can build your own model or modify one of ours. No cost or obligation.

special offer free training

For a limited time we will also include free training. **Space is limited so act now to avoid disappointment.**

Call Rick Crawford at (619) 457-9681 to reserve your place.

free trial-learn the reasons for the broad and growing popularity of SIMSCRIPT II.5—no cost or obligation

special offer-return the coupon today and we will include one free course enrollment worth $850

Name

Organization

Address

City State Zip

Telephone

Computer Operating System

Return to: IEEE COMP

CACI

3344 North Torrey Pines Court
La Jolla, California 92037

Or, better yet,
call Rick Crawford at (619) 457-9681

©1985 CACI, INC.-FEDERAL

COMPUTER

June 1986 Volume 19 Number 6 (ISSN 0018-9162)

FEATURE ARTICLES

6 Guest Editors' Introduction

James T. Cain and *Ronald G. Hoelzeman*

The Education Activities Board of the Computer Society seeks to provide sound guidelines for the education of the computer professional.

10 Education and America's Industrial Future

Lawrence P. Grayson

America's industrial strength is on the wane, and to recapture its former preeminence in the world economy it must continue to improve its educational system.

20 Design Education in Computer Science and Engineering

Taylor Booth, Tom Brubaker, Tom Cain, Ron Danielson, Ron Hoelzeman, Glen Langdon, Dave Soldan, and *Murali Varanasi*

Design must continue to be the major aspect of an engineering education in the computer area.

32 Using Low-Cost Workstations to Investigate Computer Networks and Distributed Systems

Mark Sherman and *Ann Marks*

Undergraduates can study state-of-the-art distributed systems by using collections of low-cost workstations.

42 Vocationalism and the Whole Man

Gerald L. Engel and *John F. Dalphin*

High-quality computer science programs may be accredited as long as they meet the spirit of the CSAB criteria.

47 Computing Sciences Accreditation Pro and Con

Ware Myers

Society's need for generalists and specialists creates conflict in curricula development.

SPECIAL FEATURES

50 Computer Science Education in the People's Republic of China

John H. Maier

It is to the benefit of the United States to welcome China both as a trading partner and as a new young member of the computer-based electronic village community.

57 The Japanese and software: Is it a good match?

Laszlo A. Belady

The key to the Japanese approach to software development is the separation of the planning and implementation phases.

62 A Visual Approach to Browsing in a Database Environment

James A. Larson

The operations of structuring, filtering, panning, and zooming can be used to browse a database.

Circulation: *Computer* (ISSN 0018-9162) is published monthly by the IEEE Computer Society: IEEE Headquarters—345 East 47th St., New York, NY 10017; IEEE Computer Society West Coast Office, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720. Annual subscription included in society member dues. Nonmember prices: available upon request. Single copy prices: members $7.50; nonmembers $15.00. This journal is also available in microfiche form.

Undelivered copies: Send to 10662 Los Vaqueros Circle, Los Alamitos, CA 90720.

Postmaster: Send address changes to *Computer*, IEEE Service Center, 445 Hoes Lane, Piscataway, NJ 08854. Second class postage is paid at New York, New York, and at additional mailing offices.

Copyright and reprint permissions: Abstracting is permitted with credit to the source. Libraries are permitted to photocopy beyond the limits of US copyright law for private use of patrons: (1) those post-1977 articles that carry a code at the bottom of the first page, provided the per-copy fee indicated in the code is paid through the Copyright Clearance Center, 29 Congress Street, Salem, MA 01970. (2) pre-1978 articles without fee. Instructors are permitted to photocopy isolated articles for noncommercial classroom use without fee. For other copying, reprint, or republication permission, write to Editor, *Computer*, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720. All rights reserved. Copyright © 1986 by The Institute of Electrical and Electronics Engineers, Inc.

Editorial: Unless otherwise stated, bylined articles, as well as products and services offered in New Products and New Applications, reflect the author's or firm's opinion; inclusion does not necessarily constitute endorsement by the IEEE or the Computer Society.

BPA

Computer Science Education

DEPARTMENTS

72 The Workstation: The Interpress page and document description language

78 Update: Computer Society confers top honors at Compcon Spring 86; Award winners for 1985

81 New Products

86 Microsystem Announcements

88 IC Announcements

89 New Literature

90 Open Channel: RISC for autos; Some euphemisms; Aphorisms for Abacus; Book review reviewed; Confessions of a used program salesman—an update; News of the Strictly Decimal Initiative

92 Call for Papers

93 Calendar

94 Career Opportunities

96 Advertiser/Product Index

96A Reader Service Card

7 Change-of-Address Form

COMPUTER

Editor-in-Chief: Michael C. Mulder, University of Portland

Editorial Board

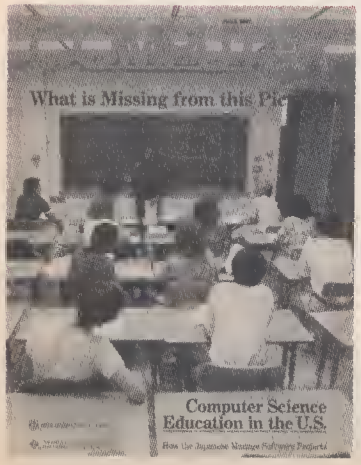
Dharma P. Agrawal, North Carolina State University
Peter P. Chen, Louisiana State University
Amrit L. Goel, Syracuse University
Jim Haynes, University of California, Santa Cruz
Ronald G. Hoelzeman, University of Pittsburgh
Franklin F. Kuo, SRI International
Francis P. Mathur,
California State Polytechnic University, Pomona
Demetrios A. Michalopoulos,
California State University, Fullerton
K. M. Mohiuddin, IBM Corporation
Leon J. Osterweil, University of Colorado, Boulder
Edward A. Parrish, Jr., University of Virginia
Arthur V. Pohm, Iowa State University
Ralph Preiss, IBM Corporation
C. V. Ramamoorthy, University of California, Berkeley
Sallie Sheppard, Texas A&M University
Harry Strong, Mitre Corporation
Richard H. Thayer, California State University, Sacramento
Wing N. Toy, AT&T Bell Laboratories
Herbert Weber, University of Bremen

Magazine Advisory Committee:
Dennis R. Allison (chair), Dharma P. Agrawal, Vishwani D. Agrawal, James J. Farrell III, Lansing Hatfield, Michael C. Mulder, David Pessel, True Seaborn, Bruce D. Shriver

Staff

Editor and Publisher: True Seaborn
Managing Editor: Richard Landry
Issue Editor: Thomas Szalkiewicz
Assistant Editors: Louise Anderson, Nancy Hays
Contributing Editor: Ware Myers
Assistant to the Publisher: Patricia Paulsen

Advertising Director: Michael Koehler
Advertising Coordinator: Carole Porter
Membership/Circulation Manager: Christina Champion
Staff Writer: Torrey Byles
Art Director: Jay Simpson
Production Supervisor: David Gaines
Design and Production: Miriam Wiegel



What is Missing from this Pic
Computer Science Education in the U.S.

On the cover

"Classroom without Computers."
Design and photography by Larry Keiser.

In the next issue

Expert Systems in Engineering.

Submissions: Submit six copies of all articles and special-issue proposals to Michael C. Mulder, Director, Applied Research, University of Portland, Portland, OR 97203; (503) 283-7433. (For additional requirements, see *Computer*, May 1983, pp. 10-11. For topics and deadlines for proposed special issues, see *Computer*'s listing in the Call for Papers section.)

IEEE COMPUTER SOCIETY EXECUTIVE COMMITTEE

President: Roy L. Russo*
IBM T. J. Watson Research Center
Route 134
PO Box 218
Yorktown Heights, NY 10598
(914) 945-3085

Vice Presidents
Publications (1st VP): J. T. Cain*
Technical Activities (2nd VP): John D. Musa*
Conferences and Tutorials: James H. Aylor
Educational Activities: Glen G. Langdon, Jr.
Membership and Information: Ming T. Liu
Area Activities: H. Troy Nagle
Standards: Helen M. Wood

Treasurer: Joseph E. Urban
Secretary: Fletcher J. Buckley
Junior Past President: Martha Sloan*
IEEE Division Directors: Martha Sloan, Ronald G. Hoelzeman
Executive Director: T. Michael Elliott*
*Ex officio member of Board of Governors

BOARD OF GOVERNORS

TERM ENDING 1986	TERM ENDING 1987
Dennis R. Allison	Barry W. Boehm
Kenneth R. Anderson	Paul L. Borrill
P. Bruce Berra	Glen G. Langdon, Jr.
Fletcher J. Buckley	Duncan H. Lawrie
Richard C. Jaeger	Susan L. Rosenbaum
Ming T. Liu	Bruce Shriver
Michael C. Mulder	Harold S. Stone
Hillel Ofek	Wing N. Toy
Edward W. Thomas	Helen M. Wood
Joseph E. Urban	Akihiko Yamada

Oscar N. Garcia, ex-officio member

PUBLICATIONS BOARD

Dharma P. Agrawal
Vishwani D. Agrawal
Dennis R. Allison
Bill D. Carroll
Michael Evangelist
James J. Farrell III
Tse-yun Feng
Lansing Hatfield
Ronald G. Hoelzeman
Sam Horvitz
Richard C. Jaeger
Willis K. King
Duncan H. Lawrie
Jack Lipovski
Ming T. Liu
Michael C. Mulder
Theo Pavlidis
David Pessel
C. V. Ramamoorthy
Bruce D. Shriver
Steve Tanimoto

J. T. Cain, Vice President for Publications

SENIOR STAFF

Executive Director: T. Michael Elliott
IEEE Computer Society
1730 Massachusetts Ave., NW
Washington, DC 20036-1903
(202) 371-0101

Editor and Publisher: True Seaborn
Director, Computer Society Press: Chip G. Stockton
Director, Conferences: William R. Habingreither
Director, Tutorials: Martez A. Camillieri
Director, Finance: Mary Ellen Curto

NEXT BOARD OF GOVERNORS MEETING:
MGM Grand Hotel
Las Vegas, Nevada
June 20, 1986, 8:30 a.m.-5 p.m.

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

President: Bruno O. Weinschel
President-Elect: Henry L. Bachman
Executive Vice President: Emerson W. Pugh
Executive Director: Eric Herz

UNIVERSITY AT BUFFALO
STATE UNIVERSITY OF NEW YORK

COMPUTER ENGINEERING FACULTY

The Department of Electrical and Computer Engineering at SUNYAB invites applications for a senior professor position in Computer Engineering. This senior faculty will play the leading role in the expansion of our program in Computer Engineering. Additional junior faculty and substantial university support are associated with the senior faculty position.

Currently there are about 30 faculty and 130 full-time graduate students in the Department. SUNYAB is in the midst of a multi-million-dollar program in upgrading its computer facilities at all levels. The Department has several VAX's, a Prime CAD/CAM system, and excellent, separate image processing facilities. Salary will be competitive.

Candidates with proven records of research and teaching should send resume, including at least three references, to Professor David M. Benenson, Chairman, Department of Electrical and Computer Engineering, Bell Hall, State University of New York at Buffalo, New York, 14260. The University is an equal opportunity, affirmative action employer.

Pride, Progress & Productivity

Three over-riding management strategies have enabled Presearch Incorporated to maintain a special niche in the fast changing world of high tech professional services:

- Unfailing dedication to improving state-of-the-art in systems, methods and equipment.
- Maintaining the responsive, "can-do" character of a small company, even though we have grown steadily in our 22 years.
- Attracting and maintaining the most qualified staff, committed to the highest professional standards.

Presearch is a valued source of professional answers to difficult questions posed by DOD, DOE and commercial clients. Our reputation for excellence extends from Information Systems Planning, Design, and Implementation in mainframe and micro environments to Software Engineering/Design and Operational Analysis. We are developing office systems in 'C,' realtime command and control systems in Ada, and are deeply committed to in-house funded R&D in Ada software engineering, embedded systems design, and software tools development.

Recent expansion has created the need for bright, energetic Computer Scientists and Engineers who are seeking to share the spirit, enthusiasm and "can-do" environment at Presearch.

For immediate consideration, call J. Wamstad at (703) 876-6404 or send your resumé in confidence to: Presearch, 8500 Executive Park Avenue, Department IE, Fairfax, Virginia 22031. An Equal Opportunity Employer. U.S. citizenship required.

Presearch

You're invited to Anaheim, California!

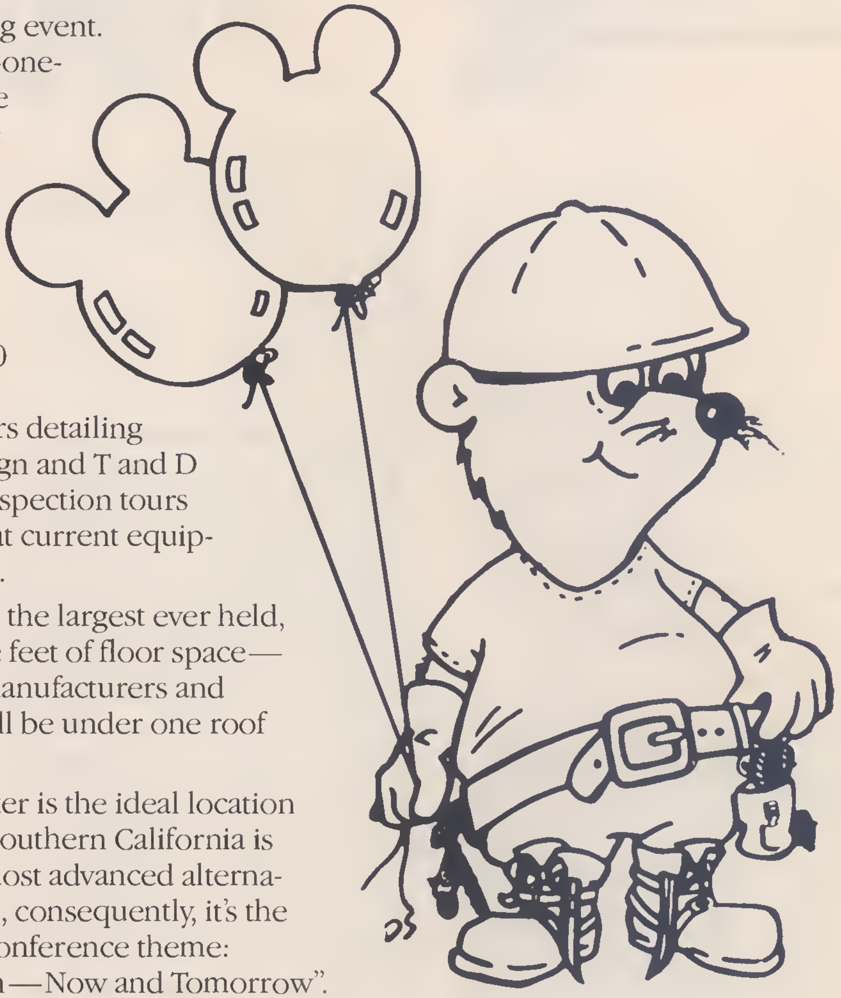
Join us for The 10th IEEE/PES Transmission and Distribution Conference and Exposition.

Don't miss this record-breaking event. Held only once every two-and-one-half years, this 10th Conference and Exposition will feature the leading edge in the industry.

The Conference will represent the latest technology and ideas in power transmission and distribution. Featured will be more than 100 technical papers presented by an array of international authors detailing their findings in research, design and T and D engineering. Plus, Technical Inspection tours will give you a first-hand look at current equipment and systems in operation.

The Exposition promises to be the largest ever held, with more than 120,000 square feet of floor space—featuring approximately 250 manufacturers and service companies. They will all be under one roof for your convenience.

The Anaheim Convention Center is the ideal location for this very important event. Southern California is home to some of the world's most advanced alternative power generating facilities, consequently, it's the perfect environment for our Conference theme: "Transmission and Distribution—Now and Tomorrow".



T.D. Mole

A "Must" for 1986. September 14-19, Anaheim Convention Center, Anaheim, CA.

10th IEEE/PES Transmission
and Distribution Conference
and Exposition

September 14-19, 1986
Anaheim Convention Center
Anaheim, California

IEEE PES

MAIL THIS COUPON FOR COMPLETE INFORMATION.

1986 IEEE/PES Transmission and Distribution Conference and Exposition
c/o Fred Klumb, General Chairman
P.O. Box 800
Rosemead, California 91770

☐ Send me complete registration materials.

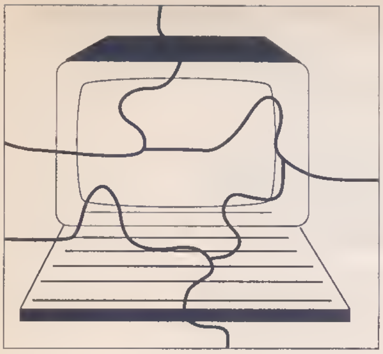
☐ Send me information on exhibiting at the 1986 IEEE/PES Transmission and Distribution Exposition.

Name ______________ Title ______________

Organization ______________

Address ______________

City/State ______________ Zip ______________



Guest Editors' Introduction

James T. Cain and Ronald G. Hoelzeman

The Education Activities Board of the Computer Society seeks to provide sound guidelines for the education of the computer professional.

The ongoing and rapid rate of technological development in the computer and associated fields continues to place a special burden on computer professionals to stay up-to-date. A solid educational foundation is essential for the lifelong learning necessary to meet this challenge. The Computer Society has a responsibility to assist its membership and the institutions that educate them in offering programs that prepare graduates for entrance into the computer science and engineering profession. The articles in this issue address some aspects of the education of the computer professional.

The nature and quality of precollege educational programs is an appropriate starting point for a discussion of the education of the computer professional. In the first article, "Education and America's Industrial Future," Larry Grayson reports on the status of precollege education in the United States. His article establishes a correlation between manufacturing productivity and the mathematics and science capabilities of the US labor force and then gives some alarming statistics relative to the capabilities of current US secondary school graduates. Hopefully, this article will provide a stimulus for all in the computer field to provide the leadership and support at both the local and national level to make the necessary changes in the US education system. Such changes are essential to higher education in general, and to computer science and engineering education in particular. Dr. Grayson notes some of the efforts of the IEEE at the Institute level to address these problems. The Computer Society is also addressing this problem through participation in the Technology in Education project sponsored by AFIPS. This project is now at the stage of testing of the prototype materials and procedures. A full report will appear in a future issue of *Computer*.

In 1977 the Society published the report "Model Curricula in Computer Science and Engineering." This report provided a comprehensive overview of the material that should be contained in a basic undergraduate curriculum. In 1981, a committee was formed to update this report, and after an extensive review, decided that all aspects of a program needed to be addressed. In 1983, The Computer Society Model Program in Computer Science and Engineering was published. This report addressed the faculty and other resource requirements of a program in addition to an update of the curriculum aspects. The Educational Activities Board, or EAB, next undertook a project to address the design component of such a program. The report on this project will be published this year. The second article in this issue, "Design Education in Computer Science and Engineering," presents an overview of that project.

Laboratories are one aspect of computer science and engineering programs that are very difficult to sustain. In 1984, the EAB initiated a project to address this issue. The highlights of this Laboratory Project are described in the Update section of this issue. In the third article, "Using Low-Cost Workstation to Investigate Computer Networks and Distributed Systems," Mark Sherman and Ann Marks of

0018-9162/86/0600-0006$01.00 © 1986 IEEE

Dartmouth address the same issue by presenting the design of a laboratory for computer networking. The article provides the highlights of an interesting educational experiment and illustrates how the laboratory can be used to teach this very important yet still somewhat amorphous topic.

The demand for educational programs in the computer area continues to exceed the ability of the higher educational system to provide quality programs. Many existing programs are marginal or substandard in quality. This presents a problem for both the prospective students and employers because of the difficulty in assessing the quality of programs. It has been recognized that professional accreditation of academic programs helps to solve this problem. The Society played a major role in establishing criteria for the Accreditation Board for Engineering and Technology, or ABET, accreditation of engineering programs in the computer area.

Programs in the computer area that are not based in engineering have, in the past, had no accreditation mechanism to assist them. This past year the Computer Science Accreditation Board, or CSAB, began operation.[1] CSAB was established by the joint effort of the Society and the ACM and provides an accreditation mechanism for this class of programs. It is now completing its first cycle of visits and evaluations. Naturally, as with any new venture, there are some areas of concern and disagreement. One class of programs for which there was concern are those based in liberal arts colleges. In the fourth article, "Vocationalism and the Whole Man," a report on two workshops, Jerry Engel and John Dalphin present the results of the some of the efforts of the Society and the ACM to address this issue.

In an effort to gain feedback on the wider issue of accreditation of computer science programs of all types, the EAB organized a panel session at Compcon Spring this year. In the fifth article, "Computer Sciences Accreditation Pro and Con," Ware Myers presents a summary of this session.

References

J. T. Cain, "Professional Accreditation for the Computing Sciences," *Computer,* Vol. 19, No. 1, Jan. 1986.

James T. Cain is currently the CS vice president for Publications, a CS representative director to and the vice president of CSAB, a member of the IEEE TAB Periodicals Committee, and a program evaluator for ABET.

Cain is an associate professor of electrical engineering at the University of Pittsburgh. His research interests lie in the areas of real time and interactive computing. He has prior industrial experience with Bell Telephone Laboratories, Bell Telephone Company of PA, and Westinghouse Electric Corporation. He received a BSEE, MSEE, and the PhD from Pitt in 1964, 1966, and 1970, respectively. He is a past recipient of a Computer Society award for the Development of the 1977 Model Curricula, 1978; the Dow Outstanding Young Faculty Award, 1977; the Western Electric Award for Excellence in Engineering Education, 1981; and the IEEE-CS Outstanding Contribution Award for Contributions to Accreditation of Computer Science Programs, 1985.

Moving?

PLEASE NOTIFY
US 4 WEEKS
IN ADVANCE

IEEE Service Center
445 Hoes Lane
Piscataway, NJ 08854

ATTACH
LABEL
HERE

- This notice of address change will apply to all IEEE publications to which you subscribe.
- List new address below.
- If you have a question about your subscription, place label here and clip this form to your letter.

Name (Please Print)

New Address

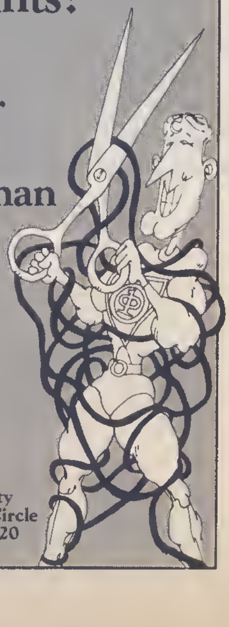
City

State/Country Zip

Ronald G. Hoelzeman is an associate professor of electrical engineering at the University of Pittsburgh. His academic interests are primarily in the areas of computer-aided engineering and computer simulation. In addition to his former position with the Computer Systems Division of the Westinghouse Electric Corporation, he has taught at the American University of Beirut. He has received Outstanding Engineering Educator Awards from the Dow Chemical Company and from the Western Electric Corporation.

Hoelzeman has held a number of leadership positions within the Computer Society and the IEEE, and is presently the Division VIII Director of the Institute.

Readers may write to the authors at the Department of Electrical Engineering, University of Pittsburgh, Pittsburgh, PA 15261.



Late Magazines?
No Magazines?
Membership
Status Problems?
No Answers
To Your
Complaints?

Let your
Computer
Society
Ombudsman
cut
through
the red
tape
for you.

Computer Society
Ombudsman
IEEE Computer Society
10662 Los Vaqueros Circle
Los Alamitos, CA 90720

Announcing TI's Second Satellite Symposium: A To Getting Started.

"To help you get started in AI we have assigned one of our top scientists... Dr. Harry R. Tennant—to explore with you the work of today's leading authorities and practical appliers, and to bring you their best advice."

Dr. George Heilmeier, Senior Vice President and Chief Technical Officer, Texas Instruments

Artificial Intelligence Step-By-Step Guide

For companies, laboratories, universities, or any organization investigating AI and the application of expert systems. A simple hookup brings it into your auditorium or conference room. On June 25, 1986

On November 13, 1985, more than 30,000 people participated in TI's first satellite symposium on AI—and 97.5% expressed interest in another one.

Announcing "Knowledge-Based Systems: A Step-By-Step Guide To Getting Started."

Symposium II is designed for managers and visionary technologists who may already be involved in AI or beginning to explore its potential. And, you need not have participated in the first symposium to benefit from this one.

Dr. Harry R. Tennant has devoted months of research surveying the work of world-class AI experts. Dr. Tennant will present the results of his investigation into the practical experiences of people directly involved in implementing knowledge-based systems.

Through on-screen interviews and interpretative commentary, Dr. Tennant will bring you much of the best advice and practical know-how that exists today.

In about five hours, you'll gain insights, perspective and step-by-step knowledge it might otherwise take months to acquire through ordinary study and trial-and-error experience.

By simply downlinking the satellite signal into your own facility, you can share this valuable learning experience with as many people as you like—at *no charge from TI!*

Learn from leading authorities and real-world experience.

You'll hear and see AI pioneers such as Dr. Edward A. Feigenbaum, noted AI author and educator from Stanford and past president of the American Association of Artificial Intelligence, and Dr. John D. McDermott, Founder and Vice President of Operations of the Carnegie Group, who turned an uncommon number of his expert systems research projects into successful applications.

Dr. Harry R. Tennant. Senior Member Technical Staff and manager of artificial intelligence research in TI's Computer Science Laboratory. Invented the concept of menu-based natural language understanding. Involved in knowledge representation systems, expert systems, logic programming, information management, and working LISP systems into the PC and other small computers. Author of "Natural Language Processing: An Introduction to an Emerging Technology." Selected as one of America's top scientists under the age of 40 by "Science Digest" magazine.

You'll also hear from executives and innovators of some of the key software tools used for development of expert systems: Thomas P. Kehler, Chairman of the Board and Chief Executive Officer of Intellicorp. James C. Williams, Vice President of Technology of Inference Corporation.

And, you'll learn from several innovative people from various companies who have successfully developed their own expert systems.

Bringing AI Symposium II into your organization is easy and inexpensive.

Almost 500 sites received the first symposium. It's neither difficult nor expensive. All you need is a satellite dish and TV monitors. If you're not already equipped, everything you need can be rented for about $1,500. We'll tell you how.

Send for full information. Now.

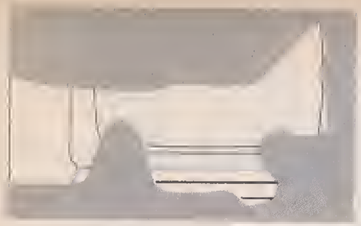
Fill out the coupon below and rush it to us for details.

TEXAS INSTRUMENTS

MAIL TO: Texas Instruments
AI Satellite Symposium II
P.O. Box 181153
Austin, Texas 78715

NAME
TITLE
COMPANY
ADDRESS
CITY/STATE ZIP

© 1986 TI 261871



Education and America's Industrial Future

Lawrence P. Grayson, US Department of Education

America's industrial strength is on the wane, and to recapture its former preeminence in the world economy it must continue to improve its educational system.

The United States is an information-based, high-technology society, with computers and other advanced technologies affecting increasingly numerous aspects of the nation's life. A particularly perceptible effect of technology is its impact on jobs and the economy. A recent study by the US Office of Technology Assessment, for example, found that in all industries considered the introduction of electronics caused production jobs—unskilled, semiskilled, and skilled—to disappear. According to the study, the typical new jobs created were for computer programmers and operators, or for service and repair personnel trained to work on the latest generation of equipment.[1] Further, of the twenty occupations projected by the US Department of Labor to grow the fastest between now and 1995, listed in Table 1,[2] almost all are related to high technology.

Although high technology will provide a minority of the nation's total job growth in the next decade or more, it also is clear that technology will impact large numbers of lower-skilled, traditional jobs in ways that are not yet fully known. It has been estimated that within about the next 15 years, 45 percent of all jobs and as many as 76 percent of the white-collar jobs will be affected by automation.[3] The US Department of Labor further estimates that by 1990 three of every four jobs will require some technical training beyond high school.

There are many academic skills that everyone should learn. High among these is an adequate knowledge of mathematics and science. Everyone needs a basic education in those subjects, not only future engineers and scientists, or even only college-bound students. A panel of employers convened by the National Academy of Sciences concluded that students not intending to go to college may not need advanced or theoretical courses in mathematics and science, but they do need a working knowledge of these subjects to perform their jobs correctly, with precision, and, most importantly, with understanding.[4] Basic skills in measuring, estimating, and calculating, as well as in the methods of scientific inquiry and measurement, are necessary in a growing number of job functions.

Professionals need an even deeper understanding of mathematics and science. A society, no matter how advanced, requires only a relatively small number of highly trained engineers and scientists to create new ideas and techniques to fuel its industries and economy. However, the development of those technologies, their production, operation, and maintenance, requires operators, mechanics, production workers, managers, and marketing personnel who have enough technical education so that they can understand, apply, and deal with the new technologies.

The major economic and social impacts of any technology come as it is diffused

US Government work not protected by US copyright.

throughout society. Although the development and sale of computers, for example, is a significant industry, their major effects occur as a result of their use to improve the productivity of existing operations and to create new products and services in factories, offices, and businesses far removed from the computer industry. This diffusion throughout society requires an educated citizenry, including consumers, lawyers, physicians, accountants, politicians, and others, who have an understanding of mathematical concepts such as probability and risk, and of the elementary principles of the physical and life sciences, so that they can resolve the broad social and environmental issues that this diffusion implies. A modern democracy requires a population that can understand the underlying technical aspects of its society in order to participate responsibly in the decisions that shape the nation and affect individual lives.

Education and industrial competiveness

Many industries now operate on a global scale, so that a nation's companies must compete against firms from other countries, both in its own and in foreign markets. The success of American businesses in this international marketplace will be determined by the quality of its work force, which to a significant degree is dependent on the education of the American people.[5] Well-educated workers are essential for technological innovation and economic competitiveness. Ill-prepared and poorly trained employees, such as those with antiquated functional skills or deficient academic attainments, ultimately make substandard contributions to a nation's industries.

Education is not the only factor affecting economic growth, but clearly it is an essential one. It is people who have new ideas, create technologies, devise processes, operate and maintain systems, and take the risks inherent in producing new, high-quality products that are necessary for the continued improvements in the standard of living and for competing effectively in international marketplaces. Education contributes to increases in a company's competitiveness, not only by improving the quality of labor, but by providing the basis for advances in technology, improvements in management practices, increases in productivity, and new product developments.

America's difficulties in competing against companies from other nations is evidenced by the problems of its automobile, steel, and aluminum industries; by its diminishing role in consumer electronics; and more recently by the growing competition from Japan and Western Europe in semiconductors, computers, and telecommunications and in industries that depend on these technologies for future growth.

Table 1. The 20 fastest growing occupations, 1982-1995.

Occupation	Percentage change in employment	Change in the number employed
Computer service technicians	97	53
Legal assistants	94	43
Computer systems analysts	85	217
Computer programmers	77	205
Computer operators	76	160
Office machine repairers	72	40
Physical therapy assistants	68	26
Electrical engineers	65	209
Civil engineering technicians	64	23
Peripheral electronic data-processing equipment operators	64	31
Insurance clerks, medical	62	53
Electrical and electronics technicians	61	222
Occupational therapists	60	15
Surveyor helpers	59	23
Credit clerks, banking and insurance	54	27
Physical therapists	54	25
Employment interviewers	53	30
Mechanical engineers	52	109
Mechanical engineering technicians	52	25
Compression and ignition mold machine operators, plastics	50	47

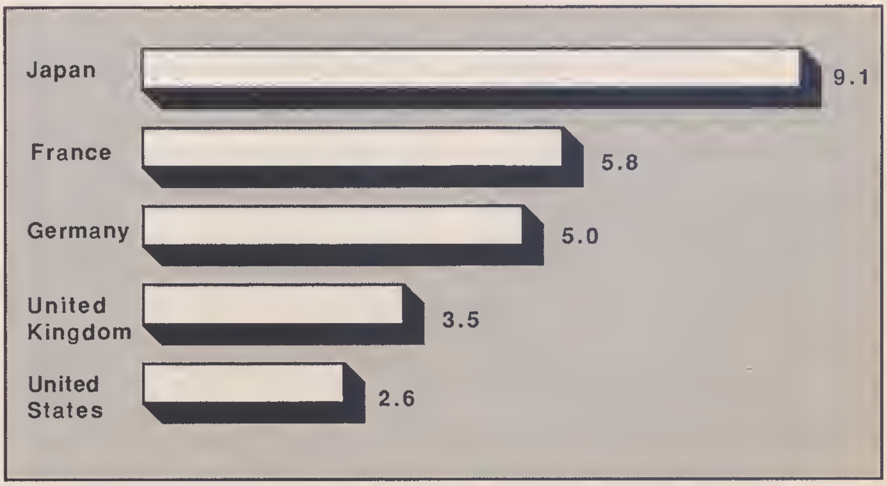


Figure 1. Manufacturing productivity growth annual rate of change, 1960-1983.

Over the last 25 years, annual growth rates in manufacturing productivity in the United States, as shown in Figure 1, have been less than that in Japan, France, West Germany, and the United Kingdom,[6] the United States' chief economic com-

Table 2. Average annual rates of change of manufacturing productivity and compensation.

	1960-73	1973-81	1982	1983	1984
Annual percentage change in productivity (output per hour)	3.0	1.7	2.1	4.3	3.5
Annual percentage change in compensation per hour	5.0	9.5	9.4	8.8	3.6

Table 3. Largest producers of cars and trucks in the free world.

Rank	1950	1960	1970	1980
1	GM	GM	GM	GM
2	Ford	Ford	Ford	Ford
3	Chrysler	Chrysler	Chrysler	Toyota
4	Studebaker	VW	VW	Nissan
5	Nash	BMC	Fiat	VW
6	Kaiser-Fraser	Fiat	Toyota	Renault
7	Morris	Renault	Nissan	Peugeot-Citroen
8	Hudson	AMC	Renault	Fiat
9	Austin	Citroen	BL	Chrysler
10	Renault	Simca	Peugeot	Toyo Kogyo
11	Fiat	Peugeot	Citroen	Honda
12	International Harvester	Rootes	Daimler-Benz	Mitsubishi
13	VW	Daimler-Benz	Mitsubishi	Daimler-Benz
14	Standard	Toyota	Toyo Kogyo	Suzuki
15	Rootes	Standard	AMC	BL

petitors. Further, as noted in Table 2, since 1960 the productivity growth rate (measured as average annual change in output per hour) has been less than the growth of hourly compensation for persons in the US manufacturing sector. As productivity growth declines in relation to wages, American firms are at a competitive disadvantage relative to those in other countries with higher productivity or lower wage structures. The greater this difference, the less able are US firms to compete both in their home and overseas markets against companies from other nations. This directly affects company sales, job availability, personal income, and, eventually, the standard of living in America.

The automotive industry is a well-publicized example of the effect of declining US competitiveness. Table 3 illustrates how the 15 largest producers of cars and trucks in the free world have changed in the past 35 years.[7] In 1950, half of these companies were American; not one was Japanese. In 1982, there were only three American companies on the list; six were Japanese. In 1985, Honda surpassed American Motors and became the fourth largest seller of automobiles in the United States. The displacement of American firms from the list means fewer jobs for American workers, and a loss of US income.

The major strides Japan has made in the growth of its industries have paralleled the development of a high level of math and science education taught in Japanese schools.[8] All Japanese high school students, for example, study the elements of probability and statistics, subjects that very few American high school students learn. Further, over 90 percent of the 18- and 19-year-olds in Japan graduate high school, while in the United States only 72 percent of this age group has completed high school. This high level of math knowledge among all graduates allows the production worker to describe problems that may occur on the shop floor in analytical terms, and allows operators, supervisors, and engineers to meet in quality circles and discuss in meaningful ways subjects such as statistical quality control. This would be impossible in the average US production operation.

The Business-Higher Education Forum in a recent study of America's economic competitiveness described the situation thus:[9] "Unless we rebuild the American economy and strengthen our educational system, it will be increasingly difficult—if not impossible—to maintain a just society, a high standard of living for all Americans, and a strong national defense."

America's performance in education

America has a strong commitment to education. Currently, the United States spends some $200 billion annually to educate 58 million students in its schools, colleges, and universities. More Americans now go to school for more years than people in any other country in history. Today, about 60 percent of US high school graduates continue to college, and about 40 percent of all 19-year-olds in the country are full-time students. The result is that the educational attainment of the civilian labor force has been rising steadily. In 1950, almost 60 percent of the civilian labor force had not graduated from high school; today, 83 percent have completed at least four years of high school. One-fourth of the present adult labor force are college graduates.

Unfortunately, time in school alone is not always an accurate measure of industrially useful, academic skills. In 1963, the United States ranked second among developed nations in the "measured skill endowments" of its labor force; by 1975, however, the United States moved to seventh place. In 1981, when 10 million people were unemployed, nearly 1 million jobs went unfilled.[10]

Americans do not necessarily need more education, but they need better education. The problem has been the quality and type of education provided, not the quantity of schooling. Unfortunately, America has allowed the quality of its educational system to decline. If children in the United

States are to benefit from and participate fully in the nation's rapidly changing society, they must receive a high-quality education, including a basic knowledge of math and science. Such learning must begin in elementary school, and continue through high school into higher education. But American schools have not been adequately preparing students for tomorrow's world.

Consider that:

- Scores on the Scholastic Aptitude Test, or SAT, which is taken by about 1 million students annually, declined 27 points in math and 47 points in verbal scores from 1963 to 1985.*
- Two-thirds of all 17-year-old students, as evaluated by the National Assessment of Educational Progress, cannot solve a mathematical problem involving several steps; 58 percent of the 17-year-olds cannot find the area of a square given one side; two-thirds cannot solve a simple math problem involving several steps; and 82 percent cannot find the area of a right triangle given the lengths of all three sides.
- There has been a steady decline in science achievement scores of 17-year-olds as measured by national assessments of science in 1969, 1973, 1977, and 1982.
- A recent study of American high schools, done for The Carnegie Commission for the Advancement of Teaching, reports that 50 percent of the students do not know that a star is more like the sun than it is like a meteor, comet, moon, or planet; 88 percent do not know that plastics are petroleum products; and 46 percent do not know that heat is produced as a result of combustion.[11]
- Today, one in six college freshmen is enrolled in a remedial reading course; one in five in a remedial writing course; and one in four in a remedial math course.
- An examination given by the state of New Jersey in 1983 to over 50,000 freshmen in all state and many private colleges and universities found that 88 percent were not proficient in ninth grade math.[12]
- Annual surveys of teacher demand show that over the seven-year period from 1979 to 1985, there was a considerable and consistent shortage of mathematics, science, and chemistry teachers in US high schools.[13]
- From 1971 to 1983, the average number of students per institution completing student teaching dropped 2½-fold in science and almost five-fold in mathematics.
- Although research shows a high correlation between verbal ability of teachers and achievement of their students, the scores on the Scholastic Aptitude Test of high school seniors intending to major in education have fallen about twice as fast as the average SAT scores, and were 23 points below the national average in verbal ability and 32 points below in mathematics in 1984.
- Further, once in a teaching position, persons scoring higher on the National Teachers Examination are almost twice as likely to leave teaching as all others with similar experience.

The problems at the elementary and secondary school levels are serious, and occur in all subject areas—in math and science, in foreign languages, in music and fine arts, in history, geography, and government, and in reading comprehension and English composition. In short, Americans are not learning the mathematics, science, and technology that is the basis for the country's future—and they are being deprived of a knowledge of the traditions, ideals, and cultural heritage that have shaped their nation.

Americans are not learning the mathematics, science, and technology that is the basis for the country's future.

International comparisons

Japan recognized the importance of education for its future 25 years ago, when its Economic Council declared:[14] "Economic competition among nations is a technical competition, and technical competition has become an educational competition." The same recognition and subsequent commitment to education persist today, not only in Japan, but in many European countries, in the Soviet Union, in the newly industrialized countries of Asia, in China, as well as in other developed and rapidly developing nations.

As a result of their emphasis on education, America's major economic competitors over the last 25 years have seen their elementary and secondary school students consistently outperform students from the United States in tests of mathematics, science, and reading. In 19 international assessments of education conducted between 1960 and 1983, the United States never placed first and was last five times among industrialized nations.[15] In most evaluations comparing the performance of students in the United States, Japan, West Germany, France, and the United Kingdom, the results have been similar—Japan is usually first, followed by the European countries, and the United States last.

In the most recent assessment of mathematics whose results are just being released, eighth grade students in the United States ranked lower, based on total score, than students from Japan, France, and England-Wales. The scores for the various categories tested are given in Figure 2. Further, among 20 nations that included both industrialized and third world nations,[16] US students ranked eighth in statistics, tenth in arithmetic, twelfth in algebra, sixteenth in geometry, and eighteenth in measurement. By comparison, Japan came in first in every category tested.

The same results have been repeated in assessments of vocational as well as academic skills. Since 1975, the United States and its major economic competitors have participated in a biannual vocational skills olympics. The abilities tested include precision machining, drafting, electronics, welding, and other skills needed by industrial workers. On average level of performance in the five competitions held between 1975 and 1983, as shown in Figure 3, the Japanese placed first and the United States last among the five nations.[17]

The free-world countries involved in economic competition are not the only ones about which to be concerned. Although the Soviet Union has not partici-

---

*In the last two years, from 1983 to 1985, math scores increased from 468 to 475, and verbal scores from 425 to 431. This is encouraging, and hopefully will continue.

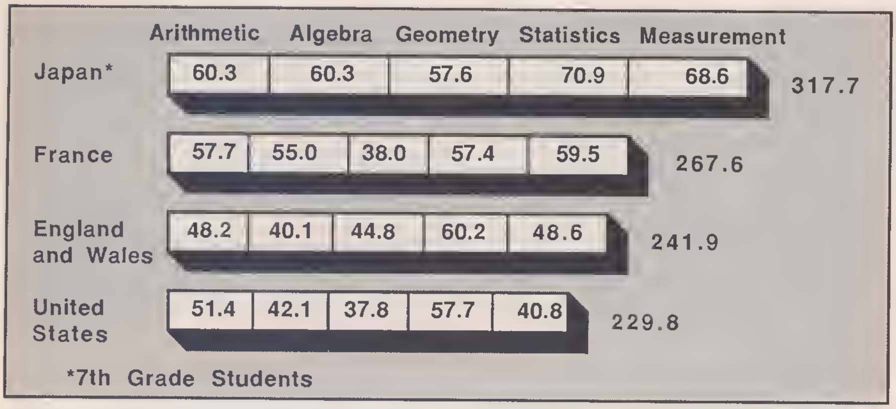


Figure 2. Second International Assessment of Mathematics, 1984. Mean scores for eighth-grade students.

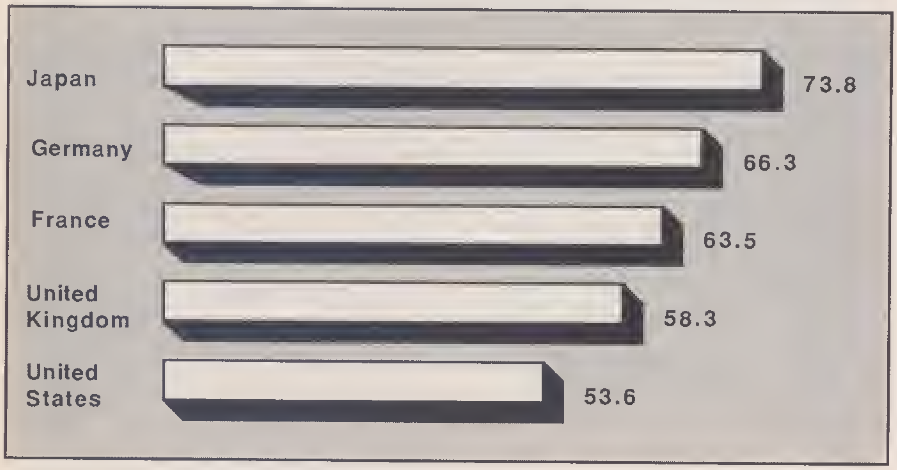


Figure 3. Average scores on Biannual International Skills Olympics, 1975-1984.

pated in the international assessments, it has developed a universal system of education that stresses math and science education to a degree even greater than Japan.

An educated population and a well-trained work force are essential for a country's economic strength. Yet, Americans have let the educational standards of its schools decline. In 1982, the National Commission on Excellence in Education released a report entitled *A Nation at Risk*. In that report, the Commission recommended that a standard high school curriculum should consist of four years of English, three years of math, three years of science, three years of social science, two years of a foreign language, and one-half year of computer science. Aside from the computer science, the recommended curriculum was commonplace 20 years or more ago for students aspiring to go to college. After the report was released, the US Department of Education did a study of the transcripts of high school students graduating in 1982. It was found that less than 2 percent, or about 1 in 50 students, completed a curriculum that demanding, in spite of the fact that more high school graduates now are continuing to college than in the past.[18]

Although only 10-20 percent of the students in France, the Federal Republic of Germany, and the United Kingdom, and 30 percent in Japan attend universities, all of these students receive a secondary education that provides them with a solid understanding of mathematics, physics, chemistry, and biology, and for some even of zoology and astronomy.[19] Since a nation's leaders are trained in universities, these nations are assured that senior officials in government, business, and industry will have an understanding of mathematics and science appropriate for informed leadership of economies dependent on science and technology for future growth. Further, students in these countries who do not attend universities, but are educated to assume mid-level technical and managerial positions or to become the skilled technicians, laborers, and operatives in industry also receive a solid education in the concepts and applications of science and technology.

The United States, in contrast, has been eminently successful in educating engineers and scientists to a high level of knowledge and creativity. Its leaders have advanced the state of knowledge through research, created new products especially in high-technology areas, and established entrepreneurial companies on a scale that surpasses that of the other countries of the world. The failing of the United States, however, is in not educating everyone to fully participate in and contribute to a society dependent on science and technology. To be competitive in today's world marketplace, government and industrial leaders who set policies, mid-level managers who must implement the policies, and the work force in general (which is largely responsible for quality and productivity gains) must have a basic knowledge of mathematics, science, and technology that is a knowledge far greater in depth and scope than most American students now are receiving.

Suggestions for the future

In order to be economically competitive on an international level, a nation must have policies and programs that enhance and improve productivity and stimulate trade in world markets; that provide incentives for the development of new and ease the transition from declining industries; and that encourage R&D, entrepreneurial initiatives, and capital formation in order to create new ideas and transform them into marketable products and services. Further, persons in or aspiring to be in the labor force must attain the training and retraining they need to adapt to the changing skill requirements of industry.

If America is to retain its preeminent, but rapidly eroding economic position in the world, it must commit itself to strengthening its educational system. American education needs more and better prepared teachers, updated curricula and teaching materials, and increased academic standards. This will be achieved only by the development of a national resolve to solve the problems of its schools, a strong commitment to excellence in education, and action by all concerned—its educational institutions, government at all levels, federal, state and local, business and industry, professional societies, and individuals. Parents, who have the primary responsibility for the education of their children, must become more involved, and must assume a greater responsibility for their children's education.

Many states and school districts recognize the problem and are acting to increase requirements for high school graduation; raise college admission standards; revise curricula; improve programs for teacher preparation and professional development; address the appropriateness of current certification requirements; and improve placement procedures, career opportunities, and rewards for teachers. Today, for example, fourteen states require three years of mathematics for high school graduation; eight states require three years of science. Although the numbers are modest, this is sharp contrast to 1980, when no state required three years of either math or science.

As numerous as these initiatives are, they are only a beginning. The United States must continue to improve its educational system, building upon the recent efforts that have been made. It must provide a high-quality education for all of its students—for those who attend college and for those who elect to work at an earlier age; for those whose parents can afford it and for those whose parents cannot; for those in large cities and for those in rural towns. It must make teaching once again an honored and desired career so that highly capable men and women will enter and remain in the profession. It must not be complacent about educational system and what its children are being thought. It must strive for excellence. The American education system must continue to be improved and revitalized so that its schools can effectively educate young American citizens to live, work, participate in, understand, and enjoy the fruits of such a high-technology society. ☐

References

1. *International Competitiveness in Electronics,* Office of Technology Assessment, Washington, DC, Nov. 1983, pp. 367-370.
2. G. T. Silvestri et al., "Occupational Employment Projections through 1995," *Monthly Labor Review,* Vol. 106, No. 11, Nov. 1983, Table 3, p. 46.
3. *Automation and the Workplace, Selected Labor, Education, and Training Issues,* Office of Technology Assessment, Washington, DC, March 1983, p. 14.
4. *High Schools and the Changing Workplace, The Employer's View,* National Academy Press, Washington, DC, 1984, p. 19.
5. D. W. Jorgenson, "The Contribution of Education to US Economic Growth," in *Education and Economic Productivity,* E. Dean (ed.), Ballinger Publishing Co., Mass., 1984, p. 97.
6. *International Comparisons of Manufacturing Productivity and Labor Cost Trends, Preliminary Measures for 1984,* News, US Dept. of Labor, report USDL 85-230, June 10, 1985.
7. Adapted from B. D. Pickel, *Ford Motor Company's Policies on Technological Change and Human Resource Development,* presented at the Conf. Education/Employment Implications of Technological Change in the Automobile Industry, sponsored by the US Dept. of Commerce and National Institute of Education, Washington, DC, Oct. 15, 1984.
8. L. P. Grayson, "Japan's Intellectual Challenge," a three-part series appearing in *Engineering Education:* part 1, "The Strategy," Dec. 1983, pp. 138-147; part 2, "The System," Jan. 1984, pp. 210-220; part 3, "The Future," Feb. 1984, pp. 296-305.
9. "America's Competitive Challenge, The Need for a National Response," *Business-Higher Education Forum,* Washington, DC, April 1983, p. 11.
10. "The Work Revolution," *8th Annual Report, National Commission for Employment Policy,* Washington, DC, Dec. 1982, p. 12.
11. Ernest L. Boyer, *High School, A Report on Secondary Education in America,* Harper & Row, Publishers, New York, 1983, p. 28.
12. New Jersey Basic Skills Council, *Report on the Results of the New Jersey College Basic Skills Placement Testing and Recommendations on Instruction and Curriculum, Fall, 1983,* Dept. of Higher Education, State of New Jersey, Trenton, Jan. 20, 1984, p. 3.
13. James N. Akin, *Teacher Supply/Demand 1985, A Report upon an Opinion Survey of Teacher Placement Officers,* Association for School, College and University Staffing, Jan. 1985.
14. Economic Council, *Report on the Long-range Educational Plan Oriented Toward the Doubling of Income,* Tokyo, Oct. 1960; cited in Nobuo K. Shimahara, *Adaptation and Education in Japan,* Praeger Publ., New York, p. 133.
15. *A Nation at Risk: The Imperative for Educational Reform,* the National Commission on Excellence, US Dept. of Education, Washington, DC, April 1983, p. 8.

(*Continued on p. 17*)

1986 TUTORIAL WEEK PROGRAMS

- Is your professional continuing education keeping pace with the needs of tomorrow?
- The Tutorial Week Programs are especially designed to assist you.
- By popular demand, we have expanded our Tutorial Week Programs from three to eight.
- Don't be left out — prepare yourself for the future!

TUTORIAL WEEK Washington '86

June 9–13, 1986

Chair: Mike Liu, Ohio State Univ.

Track I
- Microprocessor Architecture: Principles & Examples
- Semicustom Design of VLSI Circuits
- Systolic Array Processing
- Adv. Microprocessors & High-Level Language Computer Architecture
- Survivable Computing Systems

Track II
- Distributed Systems Design
- Distributed Systems: Specification, Design and Verification
- Synchronization, Resource Allocation & Broadcasting in Distributed Systems
- Principles of Message-Based Operating Systems
- Distributed Database Applications

Track III
- Data Communications
- Digital PBX: Integrated Voice/Data-Concepts, Technology and Products
- LANs, PC Networks and Workstations
- Local Computer Networks: Software Applications
- Performance Analysis of Local Area Networks

TUTORIAL WEEK Los Angeles '86

August 18–22, 1986

Chair: Masahiro Tsuchiya

Track I
- Knowledge-Based Systems
- Advanced Topics in Expert Systems
- Prolog and Knowledge Information Processing
- Knowledge-Based Systems for Software Development and Maintenance
- Computers for AI Applications

Track II
- Computer Communication Networks—Design and Analysis
- Distributed Systems
- Small Computer Local Area Networks
- Network Interconnections and Gateways
- Computer Security

Track III
- Interactive Software Development Environments
- Automated Tools for Software Engineering
- Software Testing
- Expert Systems and Intelligent Agents
- Ada*

TUTORIAL WEEK Boston '86

September 8–12, 1986

Chair: Ted Bonn

Track I
- Software Quality Assurance—A Practical Approach
- Software Configuration Management I
- Software Configuration Management II
- Quantitative Approach to Software Management and Engineering
- Ada Environments

Track II
- Tools for Building Expert Systems Development
- PROLOG and Language Processor Prototyping
- The Management of Expert System Development
- Knowledge-Based Systems for Software Development and Maintenance
- Computers for Artificial Intelligence Applications

Track III
- Networking PC's
- Fault-Tolerant Distributed Operating Systems
- Performance Analysis of Local Area Networks
- Tools for Developers of Distributed Software Systems
- Query Processing in Relational Database Management Systems

Track IV
- Compilers for Supercomputers
- Vector Processing
- Work Station Technology
- Semi-Custom Design of VLSI Circuits
- Systems Performance Gains with Parallel Processing Techniques

See future issues of Computer Magazine for details on the following events:
TW Research Triangle Park '86, October 17–20, 1986
TW San Francisco '86, (previously known as TW West), November 17–21, 1986
TW Austin '86, December 15–19, 1986

For further information return this form to:
Director of Tutorials, IEEE Computer Society
1730 Massachusetts Avenue, NW, Washington, DC 20036-1903
Telephone: (202) 371-0101

Name: ______________________

Organization: ______________________

Address: ______________________

City/State/Zip/Country: ______________________

*Ada is a registered trademark of the U.S. Government, Ada Joint Program Office

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

IEEE COMPUTER SOCIETY

16. F. Joe Crosswhite et al., *Second International Mathematics Study,* Summary Report for the United States, May 1985, Figures 23-27, pp. 121-125. Ian D. Livingstone, *Perceptions of the Intended and Implemented Mathematics Curriculm,* June 1985, Table A5, p. 36. Both reports have been prepared under contract no. OE 300-83-0212 for the National Center for Education Statistics, US Dept. of Education, Washington, DC.
17. Harold Lewis, *Report on Participation in the International Skills Olympics,* Vocational Industrial Clubs of America, 1983.
18. *How Well Do High School Graduates of Today Meet the Curriculum Standards of the National Commission on Excellence?,* bulletin, no. NCES 83-233, National Center for Education Statistics, Washington, DC, Sept. 1983.
19. L. P. Grayson "Engineering Education Around the World," *Spectrum,* November 1984, pp. 65-71.

Lawrence P. Grayson is a member of the US Department of Education, where he has served as Institute Advisor for Mathematics, Science and Technology at the National Institute of Education, and as the organizer of a major study of education in Japan. He holds a PhD in electrical engineering from the Polytechnic Institute of Brooklyn, has taught at The Johns Hopkins University and at Manhattan College, and worked at IBM before joining the federal government.

Grayson is chairman of IEEE's USAB/EAB Precollege Education Committee and was Vice Chairman of the Educational Activities Board; served as vice president and member of the board of directors of the ASEE; served as the US representative to the Education and Training Committee of the World Federation of Engineering Organizations; was chairman of UNESCO's International Working Group on Curriculum Design in Engineering Education; was a member of the US National Commission for UNESCO; and is a Fellow of the IEEE, the ASEE, and the American Association for the Advancement of Science. He is a registered professional engineer in the State of Maryland.

Readers may write to the author at the US Dept. of Education, Office of the Assistant Secretary for Educational Research and Improvement, Washington, DC 20208.

Software Professionals

NCR

Design and implement new communications network advances.

NCR Engineering and Manufacturing in Dayton is a pacesetter in the development of intelligent terminals and branch automation systems for banks, savings institutions, and other financial markets.

To keep pace with rapidly advancing software technology, we seek highly qualified professionals for key roles that offer high visibility and new technical challenge. Team up with NCR's leadership in one of the following capacities:

Software Systems Architect

In our Technology and Systems Architecture Group, you will lead the definition and design of distributed systems architecture involving intelligent workstations. Requires a BS in Computer Science or Engineering backed by experience in software design/implementation, especially in systems utilizing LAN technology. Experience in API development based on OSI model is essential. Also required is experience with PC's, the UNIX* operating system, distributed file systems and distributed error recovery.

Senior Software Analyst

In our System Software Development Group, you will implement transaction processing and distributed sysems based on state-of-the-art technology. We will look to you to act as field consultant for communications and system software for current and prospective customers. Requires a BS in Computer Science or Engineering backed by 5 years of experience in communications or transaction processing software. Knowledge of 'C' and M68000 assembler language, UNIX*, NCR TMX, SNA, X.25 and LANs desirable.

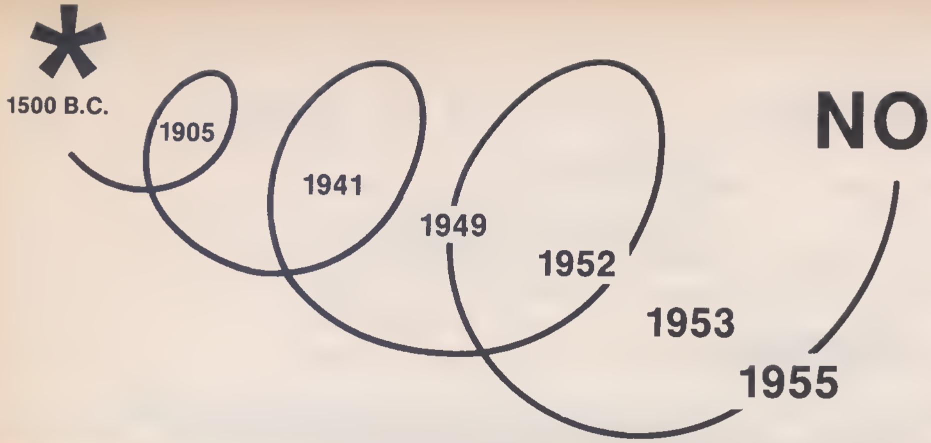
As an innovative $4-billion company, NCR offers a competitive salary, outstanding company benefits and advancement opportunity. To find out more, please send your resume and salary history indicating position of interest to:

Ms. Jayne Rohl, Dept. 118M
Engineering & Manufacturing Dayton
NCR Corporation
Dayton, Ohio 45479.

When you know where we're going, you'll choose NCR.

An equal opportunity employer.

*UNIX is a trademark of AT&T Bell Laboratories.



NOVEMBER 2-6, 1986

Explore the Knowledge-based Society During *the* Professional Conference

Really *nine* special conferences in one. Just to highlight one of the nine tracks covered at FJCC '86—

- Education
- Software Systems
- Artificial Intelligence
- Supercomputing
- Algorithms
- Modeling/Measurement
- Computer Design
- International Developments
- Operating Systems/Data Bases/LAN'S

FJCC '86 Presents Education

Major changes have taken place in education technology in the last decade. Lionel Baldwin, President of National Technological University, has put together a track that shows you what is in place and what can be done. The NTU presentations describe its ambitious offering of graduate education through satellite broadcasts of lectures direct to corporate sites. Fred Mowle and Sartaj Sahni will describe this program, and to show its effectiveness they will deliver their presentation by satellite to the FJCC. Computers, too, have had a major impact on how education is delivered. Arthur Melmed's session reveals new computer applications for elementary and high schools, colleges, and professional education.

The blackboard and chalk is giving way to new technology, and you'll see it at the FJCC '86.

* Do you Recognize the Famous Dates?

Answers to the Date Challenge

1. **1500 B.C.**—The earliest evidence of the existence of the elements of the decimal system is found in Crete.
2. **1905**—Albert Einstein Formulates the Special Theory of Relativity, establishes the law of mass-energy equivalence, creates the Brownian Theory of Motion, and formulates the Photon Theory of Light.
3. **1941**—Konrad Zuse and Dr. Schreyer complete the first fully operating electromechanical binary computer—the Z3—using a punched tape program. This system later became the control mechanism used in the V1 and V2 rockets by Germany in World War II.
4. **1949**—EDSAC, the first electronic computer to be built at Cambridge, went fully operational.
5. **1952**—Professor Hartree of Cambridge University observes that computers would "never be used for business purposes with any degree of success", and thus set back commercial acceptance of computers in the UK behind that in the US and western Europe.
6. **1953**—The Leo I computer, built by the Lyons organisation—a UK bakery manufacturer, became operational after four years of development. The descendents of this system formed the basis of a series of computers offered by ICL, who later acquired the Leo Computer Company formed by Lyons.
7. **1955**—IBM develops the Autocoder and FORTRAN programming languages.
8. **1986**—The FJCC '86, of course!

PROFESSIONAL EDUCATION PROGRAMS

The FJCC '86 **Professional Education Programs** will provide computing professionals with the opportunity to enhance their knowledge and skills. To accomplish this, an extensive program of one and two day courses will be held on November 2 and 3. In addition to traditional lecture-style presentations, there will be hands-on workshops where attendees will use computer tools and techniques.

JOIN US AT FJCC '86
INFOMART—Dallas, Texas

The Fall Joint Computer Conference '86 will be the forum for ACM's and IEEE Computer Society's Annual Conferences in 1986.

A Special Invitation to Attend FJCC '86 From The President of IEEE Computer Society

"The IEEE Computer Society is proud to combine forces with ACM to hold the 1986 Fall Joint Computer Conference. Two years ago we decided to co-sponsor a major industry event and selected a team to conduct a meeting that would replace our individual annual conferences. That team, headed by Stan Winkler, Harold Stone and Toni Shelter, has implemented concepts that far exceeded our expectations.

FJCC '86 has the key ingredients for a successful conference: a strong technical program covering the major topics of importance today, hands-on workshops that enable you to acquire a working understanding of new ideas, and an exhibits program which has attracted key vendors to use FJCC '86 to launch new product announcements.

Join us and enjoy the fruits of our labor!"

Roy L. Russo

Roy L. Russo
President
IEEE Computer Society

For further conference information, return the coupon or contact:

- Conference
  Dr. Stanley Winkler
  FJCC '86
  1730 Massachusetts Ave., N.W.
  Washington, D.C. 20036-1903
- Program
  Dr. Harold S. Stone
  IBM T.J. Watson Research Center
  P.O. Box 218
  Yorktown Heights, NY 10598
- Professional Development
  Toni Shetler
  TRW Systems Division W1/4454
  7600 Colshire Drive
  McLean, VA 22102
- Exhibits
  Tim Durkin or David McKeever
  INFOMART
  1950 Stemmons Freeway
  Dallas, TX 75207
  (214)746-3524 or (214)746-3542

Return to: FJCC '86, 1730 Massachusetts Ave., N.W., Washington, D.C. 20036-1903

☐ **YES. I'd like to attend FJCC '86** and be where the action is on November 2-6, 1986 in Dallas. **Please send me more information** about the conference and professional development programs.

☐ **Please contact me** regarding opportunities for exhibiting at FJCC '86.

Name ______________________ Telephone ____________

Affiliation ______________________________

Street Address ______________________________

City/State/Country ______________________ Zip ____________

JUN

Design Education in Computer Science and Engineering

Taylor Booth, University of Connecticut
Tom Brubaker, Colorado State University
Tom Cain, University of Pittsburgh
Ron Danielson, Santa Clara University
Ron Hoelzeman, University of Pittsburgh
Glen Langdon, IBM
Dave Soldan, Oklahoma State University
Murali Varanasi, University of South Florida

Design must continue to be the major aspect of an engineering education in the computer area.

There is a basic design methodology common to all areas of engineering, but for every branch of engineering there are some unique aspects to the design methodology. Design in the computer science and engineering field is dominated by the processing of information rather than of physical substances or energy. Design methods in computer science and engineering, or CSE, are determined by the different kinds of and means of conveying information. The algorithms that process information span the hardware-software spectrum and bridge several conceptual levels.

Beginning in undergraduate school, the process of becoming competent enough to take full responsibility for designing a major system takes an engineer many years to achieve. Every institution offering an undergraduate engineering program should ensure that its students develop design capabilities. How this is done is the responsibility of the faculty of each specific program. The Accreditation Board for Engineering and Technology, or ABET,[1] criteria require that the curriculum contain one half year of engineering design. This requirement can be met by courses at all levels of an engineering program, but one senior-level course must be predominantly an integrated design course. The design component of the curriculum must include some of the following features:

- Development of student creativity,
- Use of open-ended problems,
- Formation of design problem statements and specifications,
- Consideration of alternative solutions,
- Feasibility considerations, and
- Detailed system specifications.

Student design projects should include a variety of realistic requirements such as economic factors, safety considerations, reliability, aesthetics, ethics, and social impact as integral components of the design process. The Model Program Report published in 1983[2] by the Educational Activities Board, or EAB, of the Computer Society outlined components of a CSE curriculum and emphasized the importance of design. In 1984 the EAB formed a task force to address the problems of

0018-9162/86/0600-0020$01.00 © 1986 IEEE

properly incorporating design into a computer science and engineering curriculum. The complete report of this task force will be published by the Society.[3] The remainder of this article is an overview of that report.

Elements of design

In the world of technology, design has been the traditional province of the engineer and differentiates the engineer from the scientist. The current ABET definition of engineering design is[1]

"Engineering design is the process of devising a system, component, or process to meet desired needs. It is a decision-making process (often iterative) in which the basic sciences, mathematics, and engineering sciences are applied to convert resources optimally to meet a stated objective. Among the fundamental elements of the design process are the establishment of objectives and criteria, synthesis, analysis, construction, testing, and evaluation. Central to the process are the essential and complementary roles of synthesis and analysis."

Analysis plays a major role in the design process and is often confused with engineering design. To fully understand the design process one must understand the relationship of analysis to design.

Engineering analysis involves two major activities: model-building and evaluation. The analysis process requires an understanding of the properties and characteristics of a problem. The first step in developing this understanding is to build a mathematical model (i.e., a mathematical abstraction) that represents the important features of the system, component, process, device, or activity to be investigated. To create such a model the engineer must have an in-depth understanding of the important physical and mathematical relationships needed to solve the problem. The accuracy of the model will depend on how well it reflects the important properties and characteristics of the problem. Design decisions made on the basis of such a model will be only as good as the quality of the representation. The completed model is then used as a prototype on which all subsequent decisions about the design can be made. The model might be used to develop a set of equations describing the relationships between the values of one or more system parameters and the behavior of the system. In another application the model might be used to predict the response of a system to various types of inputs.

Although engineering analysis is an integral part of the engineering design process, many other factors must also be considered when undertaking a design. An engineer must consider a number of possible solutions to a design problem before selecting the one that "best" meets the design criteria. Some of these criteria will involve technical considerations while other criteria will involve economic and social ones. A good engineering designer must have a high degree of technical competence and an in-depth knowledge of appropriate analytical methods.

Creativity is also a part of the design process as seen when the designer comes up with several alternative solutions to a design problem. The designer then employs analytical techniques to evaluate these alternatives and select the one that best meets the design requirements. The important steps that make up the design process are summarized in Table 1.

The remainder of this article discusses ways to integrate these components of the design process into an undergraduate computer science and engineering program.

Design in the curriculum

Planning is the key to the development of a curriculum that presents an integrated view of the design process. The goal is to make design activities progressively more challenging as the student proceeds from introductory to senior-level courses. In

Table 1. Major steps in the design process.

Step	description
Define requirements	Establish a set of requirements that must be satisfied by the system. Indicate input and output of system and the operations to be performed by the system. Set any performance or economic constraints.
Develop specifications	Use system requirements to develop a set of system specifications that indicate exactly what must be accomplished by the system being designed. Specify the acceptance tests to be met by the solution.
Preliminary design	Using the requirements and specifications, develop models to represent a number of possible solutions to the design problem. List advantages and disadvantages of each solution. Select one or two preliminary designs for investigation and evaluation.
Intermediate design	Expand the preliminary designs to show the major organization and structure of the proposed system. Evaluate the refined designs and eliminate any design that is shown to be unacceptable. Use models and analysis methods to assist at this stage.
Detailed design	Develop a detailed architectural design of the proposed system. Evaluate the system to see if it will satisfy the specifications and requirements. If not, return to previous steps. Investigate prototypes of subsystems if further information is needed.
Implementation	Reduce architectural design to hardware or software that will be used to realize the system. Carry out tests to verify operation of each subsystem.
Verification and acceptance	Conduct tests on overall system to prove that it satisfies system specifications and requirements.

A typical introductory course would introduce the fundamentals of the computing process.

particular, the design tools, the documentation requirements, and the design methods presented to the students should be consistent throughout the curriculum. The planning process requires both globalwide, or curriculum-wide, planning and individual course planning. Global planning selects the point in the curriculum where major topic areas or design concepts should be introduced and identifies common material and support facilities that must be available to each student. Individual course planning must consider the details of the design topics that are to be included in addition to how the course fits into the curriculum. The initial output of the global planning process is a set of requirements for each course that include:

- The major topics covered,
- The level of design activity demanded of the students, and
- The design tools and concepts to be introduced and/or used.

In addition, the faculty should agree on common teaching or learning material, such as documentation standards, preferred programming languages, design support software, reference material, and laboratory support facilities. Although each professor may present the design process differently, each individual course must include the design content specified in the global plan. This plan should culminate in at least one significant design project in the senior year. The introductory courses in an engineering curriculum are designed to build a solid foundation in mathematics, the basic sciences, and the engineering sciences. Similarly, introductory computer science and engineering courses provide the foundation for more advanced courses in the hardware, software, and application areas. A typical introductory course would introduce the fundamentals of the computing process.[2] Such courses are ideal to start introducing the process of engineering design. Normally this course is supported by a laboratory in which a structured language is used to carry out implementations of the algorithms and the other computational activities discussed in the lecture portion of the course. Case studies should be used to illustrate how different computing problems can be solved. Experience has shown that software courses provide an excellent vehicle for introducing design content. A systematic approach to software development provides an excellent introduction to the design process. In the supporting laboratory, higher level software procedures can be used to speed up the implementation process. This makes it possible to explore meaningful design activities throughout the term. This is in contrast to most other areas of engineering, which require substantial preparation before the design process can be treated. For example, database system design problems provide a rich source of case studies for introductory computing courses. Such problems involve the need to understand the input/output process, the selection of internal data structures, different data management techniques, and the trade-offs that must be considered in selecting the best algorithms to carry out these tasks. Some typical problems in the area include the design of a simple bibliographical management system, the design of a dynamic telephone directory, or the design of an address book. These first efforts at carrying out a design must be closely monitored by the faculty member teaching the course. Executing a design can be very frustrating to a student. Encouragement and guidance are needed to make sure that each step of the design process is mastered and to provide helpful suggestions at critical points.

The initial courses typically concentrate upon the algorithmic nature of software design, and are followed by courses in hardware design. Each student should complete one or two design problems of moderate difficulty that involve both hardware and software concepts by the end of the introductory course sequence.

Courses with intermediate levels of design content introduce the student to more advanced principles and techniques and provide an opportunity to apply this material to a variety of realistic design problems. The goal of this intermediate level is to demonstrate and reinforce the fact that design activities have a common foundation in the algorithmic process and can be applied to any algorithmic, computational, or information processing problem—be it hardware, software, or an integrated hardware-software system. At this intermediate stage the conceptual levels involved run the gamut from complex software systems through the software-hardware interface, to computer design and implementation. The design stage includes medium-sized projects. The following major areas should be considered:

- Fundamentals of designing computers and software systems,
- Hardware-software trade-offs,
- The use of CAD techniques to support the design process, and
- Documentation skills.

Before proceeding to advanced courses, students should have developed confidence in their ability to carry out design assignments. Students should also have a good understanding of the interrelationships of hardware and software, and the factors that must be considered in choosing between the two when carrying out a design. Additional objectives include providing more laboratory experience, helping students understand the economic factors influencing the selection of a particular design as well as the cost of the design activity itself, familiarizing students with different metrics such as performance tests to describe the quality of a design, and finally developing their oral and written communication skills.

Advanced level design courses must introduce the students to the more advanced principles of design and give them the opportunity to jointly solve realistic large design problems using a team approach.

Students in advanced design classes should:

- Integrate hardware and software, and explore the abundant design trade-offs (microprocessor courses are ideal for this);
- Use design tools extensively;
- Work for extended periods (several weeks) in a design exercise with a group of two or three students;

- Document designs for several nontrivial projects; and
- Give oral and written presentations of the results of a design exercise.

All design projects, especially in upper division courses, should reinforce concepts being taught in class, as well as provide design experience. Course projects are usually narrower in scope than real-world design projects. However, course projects should be of sufficient complexity to challenge students to prepare good designs and force them to find solutions for problems not well-understood when the design activity begins.

Student's motivation is enhanced when they are asked to undertake design projects that represent a realistic challenge and are representative of current technology. For example, a student design team might be asked to carry out the development of a computer network to interconnect a number of personal computers, or to undertake a software simulation to support the computer-aided design of logic networks. Appendix A of the full report provides several examples of such projects.[3]

A problem with most course-related projects is that the designs do not require the use of a sufficiently broad range of skills and tools within the CSE area. Many schools require a "senior design project" of one or more academic periods that brings together concepts and tools from several courses in a single large design project involving nearly all aspects of the design process.

The faculty is responsible for designing the curriculum so that students have a range of topic areas from which the design project can be selected. Special design projects are often recommended that are related to the current research activities of the faculty or are supported by local industry. Design problems of a routine or tedious nature should be avoided.

Resource requirements

The design component of a quality undergraduate engineering program requires a significant commitment of resources by the institution. Scarcity in resources for design has the effect of discouraging students, faculty, and for that matter practicing engineers to the point of diminishing their creativity. Hence, a shortage here will have greatest negative impact on this aspect of a CSE curriculum. It is essential that the institution plan for providing, maintaining, and renewing the required resources. Plans should be made for:

- Sufficient faculty time,
- Support personnel,
- Laboratory facilities, and
- Library facilities, component specifications, and reference manuals.

More than any other single item, faculty time is the most critical factor in providing adequate design content in any program. The department must plan on allocating sufficient faculty time for program development, laboratory development, working with students, and the development of realistic design projects. Here are several areas where this extra time is required:

- Increased faculty guidance and advice to each student;
- Increased grading time for individual and detailed evaluations and comments on projects;
- Development of new design projects each time the course is taught;
- Creation of necessary lab facilities and design support tools; and
- Supervision of the teaching assistants.

Engineering faculty, as members of a professional school, must be evaluated to a significant extent on their current professional activities, which should include a major design component. Faculty members who have demonstrated design experience must teach and develop the courses with large design content. Graduate assistants are adequate to support the design courses, but should also be limited to those who have such experience.

Courses that have major design content must allow for close faculty-student interaction. Introductory courses may involve a common lecture section with a number of smaller design sections limited to 15 students. Although some of these sections may be taught by experienced graduate assistants, the faculty member in charge of the course should also teach one of the design sections. In the more advanced levels, design courses should be limited to 12 students to allow for individual meetings with project teams, to accommodate the diversity of individual projects, and to allow for detailed commentary on the written materials submitted by the students during the term.

The effective use of faculty requires support personnel.

The effective use of faculty requires support personnel. Design laboratories must be supported by an adequate number of technicians and, for large departments, a facilities engineer. The day-to-day operation and maintenance of the laboratory should be the responsibility of these support personnel under the supervision of a faculty member. Together, they must work as a team to guarantee that the needed equipment and supplies are available. As new projects are developed, it will be necessary to carry out enough preparatory evaluation of the proposed project to make sure that the project is within the capabilities of the available equipment and facilities. This work could be done by an experienced graduate assistant if a facilities engineer is not available. Experienced systems programmers must be available to support and maintain the software design tools associated with design courses (both hardware and software design). Preferably, each different operating system environment will have its own system programmer. Graduate students may be used for supplemental programming support, but do not provide enough continuity to be the only system programming resource. In light of their professional development and research requirement, faculty should never have system programming responsibilities.

Design laboratories require a major commitment by the institution. Laboratory resources are required for comprehensive projects that are complete design efforts, such as an implementation. The laboratories should be scheduled on an open basis to provide ample opportunities

(*Continued on p. 26*)

VLSI and Computers

COMP EURO 87

HAMBURG · MAY 11–15, 1987

Main Conference Topics:

1. The Impact of VLSI on Computers

- Impact on Computer Architecture, Design and Construction
- VLSI in Terminals and Personal Computers
- VLSI in Sensing and Control Applications

2. The Influence of Computers on VLSI

- Computer–Aided Design: Design Languages, Modelling, Testing
- Hard- and Software Tools for Design, Manufacturing and Test
- Device Modelling and Simulation

3. Microelectronics and VLSI: Status and Trends

- VLSI Logic and Memory Technology
- VLSI Packaging
- Microprocessors and Wafer-Scale Integration
- Production and Market Trends

4. Computer-Systems: Status and Trends

- Intelligent Workstations
- Computer Networks, Local Area Networks: LANs
- Supercomputers, Array- and Pipeline-Processors
- Special Computers: Graphics, Database, Robotics, etc.

5. Special Course: Designing Systems with VLSI-Components today

- Presentations & Demos of Cohesive and Currently Available Design Methods and Tools
- User Experiences
- Potential and Limitations of Current Solutions

Presentations
Poster Sessions
Tutorials
Exhibition
Social Program

Author's Schedule:

- Notice of intent: July 1, 1986
- 300 word summary: August 15, 1986
- Author notification: October 1, 1986
- Camera-ready copy: February 1, 1987

SPONSORS:

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS INC

IEEE COMPUTER SOCIETY

GESELLSCHAFT FÜR INFORMATIK

VERBAND DEUTSCHER ELEKTROTECHNIKER

Lufthansa Official Carrier

COOPERATING:

- IEEE Electron Devices Society
- IEEE Circuits and Systems Society
- IEEE Industrial Electronics Society
- IEEE Components, Hybrids, and Manufacturing Technology Society
- Euromicro, The European Association for Microprocessing and Microprogramming
- NTG, Nachrichtentechnische Gesellschaft
- SI, Schweizer Gesellschaft für Informatik
- VDE/VDI Fachkreis für Mikroelektronik

✂ - ✂

Please send further Information and Program booklet to

(Please use typewriter or write in capital letters!)

Title, Initials, Family Name

Company, University, etc.

Address

Address

City, State, Country, Postal Code

CompEuro '87:

Prof. Dr. W. E. Proebster
c/o IBM
P.B. 80 08 80
D-7000 Stuttgart 80
W-Germany

Membership in the Premier Computing Society Gives You The Leading Edge

acm

*The Association for Computing Machinery, **ACM,** is the largest and oldest educational scientific society in the computing field. **ACM** is the premier computing society in every sense of the word; its constitutional purposes have your interests at heart: ☐ to advance the sciences and arts of information processing and promote the free interchange of information; and ☐ to develop and maintain the integrity and competence of information processing professionals. ☐ In other words, **ACM** will give you all the technical ammunition possible to give you the leading edge. ☐ Why not join now? Use the coupon below or call our toll-free number.*

Enjoy these three super membership bonuses if you join now:

FREE: a subscription to ACM's flagship publication, *Communications of the ACM* – the magazine that gives you the leading edge every month.

FREE: one ACM Special Interest Group (SIG) membership (for the remainder of your membership year) if you subscribe to an additional ACM journal; for example if you join ACM you'll have a chance to sample SIGART on artificial intelligence. Join now and you'll receive a special "FREE SIG" packet.

FREE: an exclusive volume, "Selected Tools: Programming Pearls and Case Studies from ACM," worth $30. The "Pearls" in this volume are fun and "they teach techniques that belong in the tool box of every software engineer." These timely "Case Studies" of the NASA and TWA computer systems provide a blend of broad information and important technical detail.

FILL OUT AND MAIL THIS FORM OR CALL TOLL FREE, 24 HOURS,
1-800-526-0359 x92 (IN NJ 1-800-932-0878)

Association for Computing Machinery
11 West 42nd Street, New York, NY 10036

acm

ACCEPTANCE

☐ YES, I accept membership in the ACM for one year. I subscribe to the Purposes of ACM (above) and understand that my membership is not transferable.

SIGNATURE ________________

☐ Please list me as a VOTING member.
(complete one qualification only)
  - ☐ Bachelor's Degree from ________
  - ☐ Equivalent level of Education from ________
  - ☐ Four full-time years of experience (explain briefly)

____

☐ Please list me as an ASSOCIATE Member. I understand that I may convert to voting membership at anytime using a "Self Certification" form.

☐ I am a student; please send me a special application.

☐ I am an overseas resident; please send me a special application with Air Options.

U

(for office use only) ☐☐☐☐☐☐☐

PAYMENT

NAME ________________

ADDRESS ________________

____

CITY/STATE/ZIP ________________

☐ VOTING/ASSOCIATE MEMBER $50.00
☐ Members of the IEEE-CS $45.00
(NOT AVAILABLE to "N9" members)
IEEE-CS Member # ________
☐ Enclosed is my check made payable to ACM, Inc.
☐ Charge to my ☐ American Express ☐ Master Charge ☐ VISA

CARD NUMBER ________ EXP. DATE __/__/__

SIGNATURE ________________

for experimentation during students' free time, to allow for student trial and error, and to allow students to successfully complete the design project.

Software design projects require interactive computer facilities with rapid response time, and that are preferably located in an area with access limited to the students in the design course. There should be adequate workstations or terminals to provide each student several hours of use per week per design course. A range of software design tools should be available to support both hardware and software design activities. Hardware projects must be supported by dependable laboratory equipment, adequate expendable supplies such as miscellaneous chips, leads, etc., and a selection of reference manuals, equipment manuals, and specification sheets.

The design process includes becoming acquainted with what other designers have done to solve similar problems. Therefore, a well-supplied library facility with an up-to-date collection of technical publications and books must be available.

Many designs involve the students selecting and/or using components, subsystems, software, and instruments. A reference area, i.e., laboratory library, close to the laboratories should have up-to-date specification sheets and reference manuals for any equipment and components available to the students. This area should also include any standards documents relating to student projects and examples of previous project reports.

The computer science and engineering field has a strong component of engineering design because of the many conceptual levels involved in information systems, from hardware components to complex software systems. Hence, design must be an integral part of the educational process from the outset.

Design education can be a costly and time-consuming activity for an institution, but it cannot be omitted or diminished in emphasis. Instead design must continue to be the major aspect of an engineering education in the computer area. The Computer Society has undertaken the task of addressing the problems of design education in the form of a report. This article has presented an overview of some of the results in that report.[3] ☐

Acknowledgments

For their contribution to the Computer Society Educational Activities Board Design Project, we gratefully acknowledge Keith Barker, Jim Howard, Mike Mulder, Larry Peters, and Grace Yeung. They played a key role in the initial phases of the project and provided valuable feedback.

References

1. *1985 Annual Report,* Accreditation Board for Engineering and Technology, 345 East 47th Street, New York, NY 10017, Oct. 1985, pp. 4-115, 170-192.
2. *Model Program in Computer Science and Engineering,* IEEE Computer Society Press, Order No. 932, Dec. 1983.
3. *Design in Computer Science and Engineering Education,* IEEE Computer Society Press (forthcoming; draft copies may be requested from Glen Langdon, IBM Almaden Research Center, 650 Harry Road, San Jose, CA 95120).

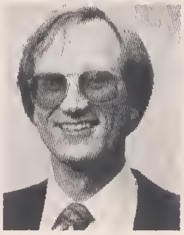
Taylor L. Booth is chairman of the Computer Science and Engineering Division of the Department of Electrical Engineering and Computer Science at the University of Connecticut, where his research includes analysis and design of high-performance software. Prior to this position, he was an analytical engineer with Westinghouse Electric Corporation in Baltimore. He received his BS, MS, and PhD in electrical engineering from the University of Connecticut in 1955, 1956, and 1962, respectively.

Booth was vice president for educational activities of the Computer Society for 1982 and 1983. He has been a major participant in computer science and engineering education development since the mid-1960's, when he was a member of the Cosine Committee of the National Academy of Engineering. He is now a fellow of the IEEE and a member of ACM and ASEE. He was also editor-in-chief of the *IEEE Transactions on Computers.*

Tom Brubaker received the PhD from the University of Arizona in 1963. He has taught at the University of Wyoming, the University of Missouri, Columbia, and is presently professor of electrical engineering at Colorado State University. His research interests include digital signal processing, digital system design, and satellite meteorology.

Brubaker is a senior member of the IEEE and serves as chairman of continuing education on the EAB. His considerable industrial experience now includes being president of MET-SAT, Inc., a company specializing in satellite meteorology and space applications.

Tom Cain and **Ron Hoelzeman**'s biographies appear at the end of the Guest Editor's Introduction, p.7.



Ronald L. Danielson is associate professor of computer science and director of information systems at Santa Clara University in California. His administrative duties involve coordinating all university computing and telecommunications activities. His research interest lie in the areas of image processing system design and application, and he has served as a consultant in those areas to federal and state governments as well at to local industry.

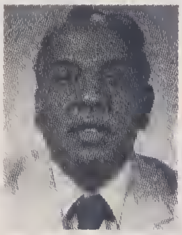
Danielson received the BA and MS degrees in mathematics from the University of Minnesota, Duluth, and Northwestern University, respectively, and the PhD degree in computer science from the University of Illinois at Urbana-Champaign. He is a member of the IEEE Computer Society and the ACM.

Glen G. Langdon, Jr. is a research staff member in the computer science department of the IBM Almaden Research Center in San Jose, California. He is vice president for educational activities of the IEEE Computer Society, and an elected member of its governing board. He was general chair of Compcon Spring 86. He received a BSEE from Washington State, an MSEE from PITT, and a PhD in EE from Syracuse.

With IBM since 1963, Langdon has worked in computer design and architecture, error detection, and data compression systems; has authored two books on logic and computer design; and coauthored a Brazilian computer design book in Portuguese. He taught for four semesters at the University of Sao Paulo, and his students did the Patinho Feio in 1972, the first computer designed and built in Brazil. He is a CSAB visitor for computer science programs.

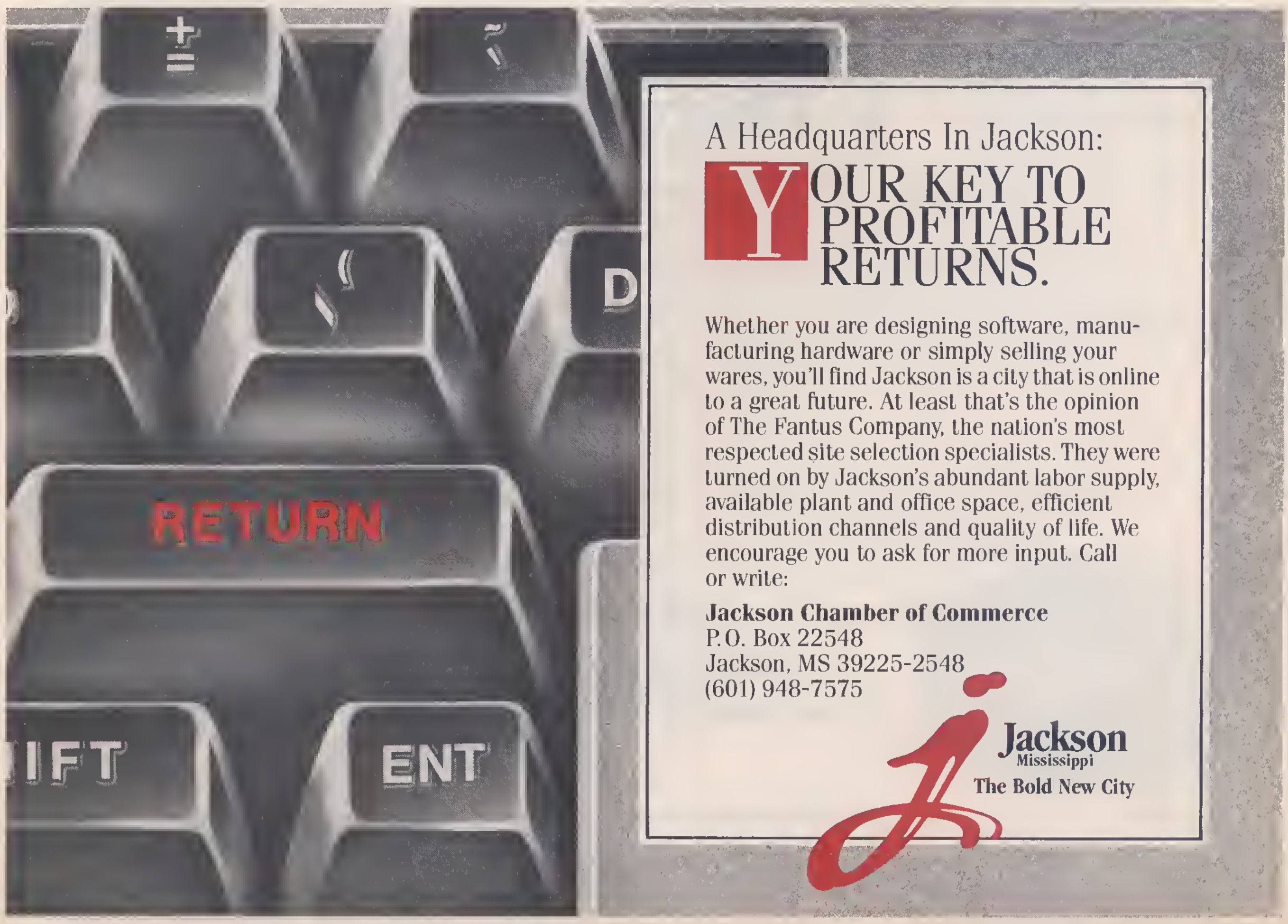
David L. Soldan is an associate professor of electrical and computer engineering at Oklahoma State University in Stillwater. His teaching is in the area of logic design and testing of digital systems. He is coprincipal investigator of a project dealing with the application of vision systems to automated manufacturing.

Soldan received his PhD degree from Kansas State University in Manhattan, Kansas. He is a member of the IEEE Computer Society and Acoustics and Speech and Signal Processing Society. He is currently chairman of the Computer Society's EAB Design Project and Curriculum Assistance Committees and is a member of the EAB's Laboratory Taskforce.



Murali Varanasi received BSc and DMIT degrees from Andhra University and Madras Institute of Technology in India. He received the PhD degree in electrical engineering from the University of Maryland, College Park in 1972 and 1973, respectively. He was with the department of electrical engineering, Old Dominion University in Norfolk from 1973 to 1980. He is currently with the department of computer science and engineering, University of South Florida, Tampa. His research interests include coding theory, fault tolerant computing, computer architecture, and image processing.

Varanasi is an active member of the IEEE Computer Society and is currently vice chair of the EAB.



A Headquarters In Jackson:

YOUR KEY TO PROFITABLE RETURNS.

Whether you are designing software, manufacturing hardware or simply selling your wares, you'll find Jackson is a city that is online to a great future. At least that's the opinion of The Fantus Company, the nation's most respected site selection specialists. They were turned on by Jackson's abundant labor supply, available plant and office space, efficient distribution channels and quality of life. We encourage you to ask for more input. Call or write:

Jackson Chamber of Commerce
P.O. Box 22548
Jackson, MS 39225-2548
(601) 948-7575

Jackson
Mississippi
The Bold New City

Reader Service Number 1

SAVE $50!

WHEN YOU ATTEND THE PREMIERE OF NCC-TELECOMMUNICATIONS '86, SEPTEMBER 8-10, IN PHILADELPHIA, PA.

As a member of a sponsoring society for NCC-Telecommunications '86, you're entitled to a special $50 discount on this year's premiere Full Conference Registration Fee. A discount that pays off double in the competitive advantage you'll gain by attending NCC-Telecommunications '86.

Why? With over 100 exhibits, expert Conference Sessions, and intensive Professional Development Seminars, you'll see, hear and examine everything that's new in telecommunications. And come away with answers to your questions and solutions to your problems.

IEEE
COMPUTER SOCIETY
MEMBER
DISCOUNT

So whether you need to know about integrated services data networks, voice recognition, local area networks, micro-mainframe linkups, video conferencing, digital network facilities, voice/data/text integration, micros vs. mainframes, and more, you'll find it at NCC-Telecommunications '86.

Register now to attend the premiere of the largest and most intensive telecommunications conference on the East Coast. Fill out and return the Registration Form on the next page. You'll save $50! And be on your way to gaining a new telecommunications advantage.

Fill out and mail the form on the next page.

You can even charge your discounted Registration Fee to your VISA, MasterCard or American Express!

NCC-TELECOMMUNICATIONS '86

"Gaining A Competitive Advantage"
September 8-10 • Philadelphia Civic Center
Philadelphia, PA

Sponsored by: American Federation of Information Processing Societies; Association for Computing Machinery; Data Processing Management Association; IEEE Computer Society; Society for Computer Simulation.

	Program Hours	Exhibit Hours
Mon.	10:00-5:30	11:00-6:00
Tues.	8:30-5:30	10:00-6:00
Wed.	8:30-12:00	10:00-4:00

NCC-TELECOMMUNICATIONS
IEEE COMPUTER SOCIETY (IEEE-CS)
September 8-10, 1986
Civic Center — Philadelphia, Pennsylvania

REGISTER BY PHONE
To obtain further Registration information or to register by phone using your credit card dial:
800-NCC-1986
(In Virginia 703-620-8955)
Telex—710-833-9037
Last day to register by phone is September 4, 1986
Lines are open Monday through Friday 9:00 am - 5:00 pm Eastern time.

Register now and save time and money! Advance Full Conference price $125. On-Site Full Conference price $150

Please Print or Type and Fill Out Completely

1 ☐ MR 2 ☐ MRS 3 ☐ MS 4 ☐ DR CHILDREN UNDER 14 NOT ADMITTED KG

1

NAME Last First MI
TITLE
COMPANY
ADDRESS
CITY STATE ZIP
COUNTRY TELEX
TELEPHONE EXT.

☐ Check here if you do NOT wish your name to be on our mailing list

2 REGISTRATION CHOICES

Please mark your selections and include total — Member Disc Prices:

11 ☐	FULL CONFERENCE PROGRAM & EXHIBITS Includes Conference Digest, Total advance registration price. On-site cost $150.	$125
33 ☐	STUDENT—Full Conference Program & Exhibits No Digest	$ 25
44 ☐	EXHIBITS ONLY — Monday through Thursday does not include Conference Program	$ 20
55 ☐	MONDAY ONLY — Conference Program & Exhibits No Digest	$ 75
56 ☐	TUESDAY ONLY — Conference Program & Exhibits No Digest	$ 75
57 ☐	WEDNESDAY ONLY — Conference Program & Exhibits No Digest	$ 75

TOTAL REGISTRATION $ ______

PROFESSIONAL DEVELOPMENT SEMINARS

A series of in-depth, audience specific seminars will be offered as a complement to the Conference Program.

Professional Development Seminar Fees for full-day ($200) and half-day ($125) are IN ADDITION to the registration fees. Continuing Education Units are available.

Call 800-NCC-1986 for more info

Watch for more info in Conference Call

3

METHOD OF PAYMENT

Make check payable to NCC '86

NO FOREIGN CURRENCY
NO PURCHASE ORDERS
NO GOVERNMENT VOUCHERS

- ☐ Check
- ☐ American Express
- ☐ VISA
- ☐ Mastercard

PAYMENT MUST BE INCLUDED WITH REGISTRATION
(Enter credit card number here)

EXPIRATION DATE ______________

SIGNATURE ______________

4 Please Check the Appropriate Blocks In Each Area to Complete Your Registration.

A WHAT IS YOUR INDUSTRY CATEGORY?
[check one]
- A ☐ Computer/Peripheral/Equip. Mfg.
- B ☐ Electrical/Electronic Component
- C ☐ Computer/DP Service
- D ☐ Software
- E ☐ Research/Development
- F ☐ Wholesale/Retail
- G ☐ Banking/Insurance/Real Estate
- H ☐ Medical/Legal
- I ☐ Government
- J ☐ Education
- K ☐ Other Business Services
- M ☐ Manufacturing
- N ☐ Utilities
- O ☐ Aerospace
- L ☐ Other ______

B WHAT IS YOUR JOB FUNCTION?
[check one]
- A ☐ President/Vice Pres./Owner
- B ☐ Manager/Supervisor
- C ☐ Engineer
- D ☐ Analyst
- E ☐ Programmer
- F ☐ Consultant
- G ☐ Other ______

C WHAT IS YOUR PRIMARY AREA OF INTEREST?
[check all applicable]
- A ☐ Computer Systems-Business
- B ☐ Computer Systems-Scientific
- C ☐ Personal Computers
- D ☐ Telecommunications Systems
- E ☐ Computer Graphics
- F ☐ Word Processing
- G ☐ Data Processing
- H ☐ Software
- J ☐ Peripheral Equipment
- K ☐ Data Communications Equipment
- L ☐ Data Storage/Memory Systems
- M ☐ Microfilm/Microfiche
- N ☐ Computer Services
- O ☐ Consulting Services
- P ☐ Supplies & Accessories
- R ☐ Telephone Systems
- S ☐ Facsimile Equipment
- T ☐ Local Area Networks

D NUMBER OF EMPLOYEES AT YOUR LOCATION:
[check one]
- A ☐ 1-19
- B ☐ 20-49
- C ☐ 50-99
- D ☐ 100-499
- E ☐ 500-1000
- F ☐ over 1000

E COMPUTERS LOCATED AT YOUR COMPANY SITE.
[please fill out completely]

micro	mini	main	CPU Manufacturer	Model #	Operating System
☐	☐	☐			
☐	☐	☐			
☐	☐	☐			
☐	☐	☐			
☐	☐	☐			

REGISTRATION DEADLINES

Deadline for MAIL-IN registration must be postmarked by **August 15, 1986**. You may register by telephone or telex through **September 4, 1986**.

Cancellations and changes must be made in writing and received by **August 22, 1986**.

You must enclose any registration materials received to obtain a refund.

MAIL TO:
NCC-Telecommunications
1970 CHAIN BRIDGE ROAD
McLEAN, VA 22109-0166
(Must include complete zip code)

SOCIETY DISCOUNT REGISTRATION MUST BE SUBMITTED ON ORIGINAL FORM

The One Leveraged Investment With a Guaranteed Return

You design and build the systems that allow information brokers to conduct business. They turn to you for guidance. But where do you turn? How do you find the knowledge that strengthens your own technology portfolio? How do you stay abreast of the most volatile market of all: the market of ideas in information science? And how do you look into the future to chart your own professional growth?

You do what 90,000 other information professionals have done worldwide: JOIN THE IEEE COMPUTER SOCIETY.

Through its periodicals, conferences, technical and standards committees, local chapters, and wide variety of books, the IEEE Computer Society brings you into direct contact with the leaders in computer technology.

And when the Computer Society talks, people listen. Its members set the industrial standards for computers, establish college curriculum guidelines, serve as expert consultants to governments, and help nations transfer technology among professional/scientific communities throughout the world.

Start acquiring the intellectual capital you need for the information age. Join the IEEE Computer Society.

IEEE COMPUTER SOCIETY

IEEE COMPUTER SOCIETY

Membership Application

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC

Schedule of Fees (tax deductible)

Membership dues and publication subscriptions are annualized to December 31. Depending upon the date your application is received by the Computer Society, pay the full annual amount or one-half the annual amount as indicated below. Membership expires December 31.

	Half-Year Mar 1–Aug 31	Full-Year Sept 1–Feb 28
1 I already belong to the IEEE and I want to join the Computer Society IEEE Member Number ____________________	☐ $5.00	☐ $10.00
2 I don't belong to the IEEE and I want to join just the Computer Society	☐ $17.00	☐ $34.00

3 I don't belong to the IEEE and I want to join both the Computer Society and the IEEE*
Full IEEE members receive *Spectrum*, *The Institute*, and many additional benefits.
(Total amount to be paid includes annual dues and regional assessment, if any)

	Half-Year	Full-Year
I reside in Region 1-6 (United States)	☐ $38.50	☐ $77.00
I reside in Region 7 (Canada)	☐ $36.00	☐ $72.00
I reside in Region 8 (Europe, Africa, or the Middle East)	☐ $34.50	☐ $69.00
I reside in Region 9 (Latin America)	☐ $31.00	☐ $62.00
I reside in Region 10 (Asia and Pacific)	☐ $31.50	☐ $63.00

*ACM members who join both IEEE and the Computer Society are entitled to a deduction of $5 off the full-year rate $2.50 off the half-year rate.

4 In addition to my automatic subscription to *COMPUTER*, I would also like to subscribe to:

	(issues per year)	Half-Year	Full-Year
IEEE Expert (3151)	4	☐ $ 5.00	☐ $10.00
IEEE Computer Graphics and Applications (3061)	12	☐ $11.00	☐ $22.00
IEEE Design and Test (3111)	6	☐ $ 7.00	☐ $14.00
IEEE Micro (3071)	6	☐ $ 7.50	☐ $15.00
IEEE Software (3121)	6	☐ $ 7.00	☐ $14.00
Transactions on Computers (1161)	12	☐ $ 7.50	☐ $15.00
Transactions on Pattern Analysis and Machine Intelligence (1351)	6	☐ $ 6.00	☐ $12.00
Transactions on Software Engineering (1171)	12	☐ $ 7.50	☐ $15.00
Journal on Lightwave Technology (4301)	4	☐ $ 5.00	☐ $10.00
Journal of Robotics and Automation (4401)	4	☐ $ 5.00	☐ $10.00
Journal of Solid State Circuits (4101)	4	☐ $ 2.50	☐ $ 5.00
Proceedings of the IEEE (5011) (available only to IEEE members)	12	☐ $ 7.50	☐ $15.00

☐ VISA ☐ MasterCard ☐ American Express

Charge Card Number ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Exp. Date Mo ☐☐ Yr ☐☐

☐ Check or money order enclosed. Make payable to IEEE.

$ ____________ Total amount remitted with this application

Check box for additional information: ☐ Student Membership Application ☐ Publication Catalog
☐ Bulk air option (two-to-six-week delivery of publications) for members in Western Europe, Latin America, and the Far East

I hereby make application for IEEE Computer Society membership and, if elected, will be governed by IEEE's and the Society's Constitutions, Bylaws, and Statements of Policies and Procedures.

MAILING ADDRESS

Full signature ____________________ Date ____________

First Name ____________ Middle initial(s) ____________ Last name ____________

Street address ____________ City ____________ State/Country ____________ Zip ____________

EDUCATION (highest level completed) Course ____________ Degrees received ____________ Date ____________

Name of educational institution ____________ Date of birth ____________ ☐ Male ☐ Female

OCCUPATION Title or position ____________

Firm name ____________

Firm address ____________

City ____________ State/Country ____________ Zip ____________

Affiliate society membership no. (if applicable) ____________ Grade ____________

REFERENCE (an IEEE member; if unknown, a managerial person who knows you professionally)

Name (print in full) ____________ IEEE Member No. (if applicable) ____________

Position ____________

Street address ____________

City ____________ State/Country ____________ Zip ____________

MAIL TO: IEEE COMPUTER SOCIETY, 10662 LOS VAQUEROS CIRCLE, LOS ALAMITOS, CA 90720–2578

C: 6/86

Using Low-Cost Workstations to Investigate Computer Networks and Distributed Systems

Mark Sherman and Ann Marks, Dartmouth College

Undergraduates can study state-of-the-art distributed systems by using collections of low-cost workstations.

A quality education in contemporary computer science requires that students gain experience with realistic systems. Many efforts at bringing empirical computer science to undergraduates focus on rather old technologies, for example, building a compiler or simulating a disk scheduler. Although efforts are being made to use some newer technologies, such as the advanced graphics of the workstations in Brown's Balsa system[1-4] and MIT's Project Athena,[5] the efforts are concentrating on teaching traditional material in a new medium. However, the medium itself—networked workstations in a server environment—is worthy of exploration by undergraduate students in a laboratory setting. At Dartmouth, we developed a Computer Network Laboratory to let students experiment with computer networks, protocols, and distributed systems. Through this article, we wish to share our experiences in designing the laboratory and give an example of how the laboratory was used in a computer network course.

Requirements for the facilities

The requirements for a computer network or distributed systems laboratory are quite different from those for facilities needed for a typical computer science course in algorithms, data structures, or compilers. When we decided to offer a network course emphasizing the experimental aspects of computer networks, we examined our campus-wide time-sharing facilities and found them inadequate. We decided that our laboratory facilities required

- Many machines,
- Reconfigurable facilities,
- Unprotected, open systems, and
- Economy and uniformity.

Each criterion is explained below.

Many machines. To investigate how a distributed system worked, we needed to physically distribute the system among several machines. Therefore each student would require some collection of machines for each assignment. As we describe in the sidebar Development Systems for Students, we allocated five machines per student. With so many machines, we also found that we needed extra machines to cover the inevitable breakdowns during the course.

Reconfigurable facilities. Because networks can use a variety of topologies, we needed to provide a way to reorganize the machines to create networks. Some exper-

0018-9162/86/0600-0032$01.00 © 1986 IEEE

iments examined contention on a bus that required many hosts to generate traffic; others measured a gateway between two networks; and still another considered only point-to-point communication without interference. Each experiment required a different configuration of machines. Further, our laboratory uses distributed development and debugging facilities. For example, a gateway experiment may require network monitors on each network in contrast to a contention experiment that needs only a single monitor. Therefore, it is essential that we be able to organize our facilities in many different ways without extensive hardware construction.

Unprotected, open systems. The access provided by an unprotected, open system is needed to support a broad range of activities. Because many of our experiments need to replace or monitor very low-level features of the operating system, we required that students have full access to any part of the machine. For example, some of our experiments were designed to measure the overhead expense of using general operating system facilities in a distributed system vs. making a special-purpose operating system. Some facilities, such as a network monitor, require writing very low level parts of an operating system that directly control the network hardware.

Economy and uniformity. We have found that if students can use a set of equipment for more than one course, and if they can use the equipment after the course has finished, there is a greater incentive to learn the details of the systems and development aids. With a greater mastery of the basic system, the students are able to concentrate their attentions on the course material rather than on the details of the particular system they are using. One way to provide the uniformity is to encourage the students to buy a machine of their own. Obviously, students' funds are quite limited, and so the machine must be economical. Thus the need for economy is not merely so that we can afford the many machines required for the course, but so that individual students can also afford to purchase equipment for their own use. A similar effect is achieved if uniform equipment is used in a curriculum: having learned, for example, Unix, in one course, a student brings his or her knowledge of how to use the editor and mail system to the next course using Unix.

Inadequacy of current systems

Unfortunately, available systems did not meet our criteria. Historically, educational institutions have relied on a campus computer utility for teaching the practical aspects of computer science. The utility is usually a centralized time-sharing system. But neither a utility nor a centralized, time-sharing system provides adequate facilities for experimenting with computer networks or distributed systems. The nature of a shared resource for use by students usually implies

- Other people rely on the utility;
- Relatively few systems are available;
- Systems cannot be reconfigured; and
- Nondeterministic and asynchronous aspects of systems are inaccessible for study.

Because other people rely on the computer utility, the computation center (or other similar authority) must ensure that the utility is available, reliable, robust, and secure. None of these aspects can be assured if a large number of inexperienced students start making kernel changes in the running system. Therefore, students using a shared facility have limited access to a system's resources and underlying hardware features. The limitations were designed to protect the system from a user and to protect one user from another. For example, a nonprivileged user cannot write and measure a true elevator scheduling algorithm for a disk drive—doing so would jeopardize other users who store files on the disk. Instead, a user is typically allowed to use synchronous calls to file system primitives that provide a high-level byte-stream abstraction. For the student, a simulator is employed to experiment with a disk scheduling algorithm. Unfortunately, such simulators usually beg the issue of what is to be demonstrated, since there is no way for most simulator writers to compare their simulators against the actual system—the simulator writers are also nonprivileged users!

Even if students were allowed to change a utility system, a typical computer utility is inadequate for a network or distributed systems laboratory. First, the students' experiments could not be adequately controlled; that is, other users of the system would perturb the results and interfere with the students' work. Second, most utilities are wired into a campus-wide network that has a predetermined configuration. The hardware and software for reconfiguring the network is generally unavailable, even if the utility's operators were inclined to let a class reconfigure a system.

Using collections of workstations

Because the centralized, time-sharing utility could not meet our needs, we were drawn toward the use of workstations as the basis of our laboratory equipment. A typical workstation provides many of the same environments available on a shared system. However, a user may arbitrarily change the underlying configuration of a workstation's software or hardware in order to experiment with alternative techniques or to demonstrate certain properties. Some features that can be investigated include new disk configurations and algorithms, network protocols, interrupt structures, processor scheduling algorithms, and memory allocation algorithms. Another feature of workstations is their economy: one can purchase several workstations in order to demonstrate distributed algorithms, to investigate system architectures that improve reliability, and to experiment with network services.

While we were designing our laboratory, Dartmouth College became an Apple University Consortium member, which allowed us to purchase Macintosh equipment[6] at a substantial discount. Therefore we could obtain the necessary equipment at an acceptable cost. Further, we believed that students would purchase large numbers of the same machine for their own use. At the same time, a decision was made at Dartmouth to use the network architecture being proposed by Apple (now called AppleTalk[7]) to connect together all the Macintoshes on the campus. Therefore we could expect to find local expertise on the construction of networks using Macintoshes. Thus we made a decision to run our laboratory on Macintoshes with AppleTalk.

Development systems for students

A successful computer laboratory needs sufficient tools to allow students to perform their work effectively and efficiently. Especially among manufacturers who offer reduced or restricted systems for "educational use," there is a widespread belief that student facilities need only minimal support. In fact, students need the most advanced tools to which they can gain access. All the usual needs to improve productivity that are asserted for commercial projects apply equally well to students—those alone should justify the tools. Because of the academic calendar, students face some unusual situations that demand sophisticated tools for their projects: exceptional learning curves, productivity requirements, and strict deadlines.

Exceptional learning curves. A typical 10-week quarter will have about eight assignments, or one per week. Since academic regulations generally prohibit assignments due dates during the last two weeks of a term (to prevent finals being surreptitiously given during the regular term), students are required to produce some relevant piece of software during their first week in class. The students must therefore learn the editor, decode compiler diagnostic messages, understand any provided packages, and interpret the documentation before they can start on the intellectual content of the first assignment. Unlike conditions in an industrial environment, students are expected to work *without* collaboration of their coworkers. Therefore each student must learn a great deal of incidental material before any real work can actually be performed. A good set of tools can greatly minimize this effort. As a simple example, a structure editor can save time by making sure that a program is syntactically correct. By the time a student gets to an advanced course, he or she understands the concept of a block and should not waste time learning that PL/1 uses the predefined identifier **do** and requires a semicolon while Pascal uses the keyword **begin** and has no semicolon. Good tools can improve the learning curve of a system and permit a student to spend time on relevant aspects of a course.

Productivity gains. A second reason for good tools is to minimize the unproductive work that a student performs. A typical student programming environment requires the student to perform source control, backup, and even the basic edit-compile-link cycle in a primitive or manual fashion. Even when the student is familiar with the system software, a student can waste enormous amounts of time verifying a particular linker switch or breaking a program into small enough units so that a compiler's symbol table does not overflow. Although this time might be considered useful if one wants to experience "real world" problems, the time is wasted in that less effort is spent learning compilers, operating systems, or computer networks.

Unfortunately, the problem here is not only with unavailable tools, but also with poorly written assignments. In examining the materials used in a variety of courses at a number of schools, we saw exercises of such breadth that realistic scaffolding for the problem was as large as the final program. Although learning how to build scaffolding is useful in general, it is irrelevant to the objectives of most assignments. Therefore instructors must take the responsibility of ensuring that sufficient supporting materials are available for assignments. Together with good tools, students can concentrate on the content of the assignments and not merely on the form required by the systems programs.

Hard deadlines. Like industry, students have real deadlines. Although a grade penalty can lure some students into meeting the weekly deadlines, it is the end-of-term deadline that causes the most problems for a student. The problems we are concerned with are not whether a particular program is written or not, but whether the student has had an adequate opportunity to study the material in the course. In industry, one can usually slip "one more week." When a term ends at a university, there are no more weeks to slip; the instructor cannot cover another lecture, extend another assignment, or require another reading. Therefore students must be able to finish their work expeditiously. The end-of-term deadline is immutable; missing that deadline usually means misunderstanding some course material. Therefore an instructor should try to provide good enough tools that expedite the completion of assignments.

Some requirements for an educational environment. A reasonable environment for software laboratories in computer science should include the following kinds of tools:

Quality editors. At the very minimum, a good screen editor should be provided. Better would be a programmable editor, such as Emacs. Still better would be a structure editor for the languages being used. Ordinary text editors with a good string matching and replacement facility, batch editors, and file-difference locators are also valuable aids.

Translators. A fast compiler with good diagnostics is needed. For large projects, a good, fast separate compilation facility is needed. For intricate algorithms, a good interpreter for the same language can help immensely. The translators should either provide or be integrated with symbolic (source level) debuggers and performance analysis tools.

Source control. Large projects are built out of many parts and will require tools to build and manage libraries. Students need a way to organize their materials and coordinate efforts among many people. A source control tool turns out to be critical when some members of a student project live on campus and some live off campus. Such people cannot effectively leave notes for one another and rely on technological aids to keep order in their chaotic system.

Mail systems. A good mail system is needed to record notes on design decisions, communicate about meetings between team members, ask for extensions from the instructor, and coordinate people with vastly different schedules who cannot attend face-to-face meetings.

Printers. Students need fast, convenient, good-quality hard copy for their work. When a computation center piles all listings for an entire day for all users on a table in random order, the students waste both their own time and computer time resubmitting print requests for lost (or unfound) listings. Three-hole punched, notebook-sized paper is far more convenient than large, fan-folded 14-inch paper. As students start to use bit-mapped displays for design documents, the printing facilities for these graphics must be provided as a matter of course, not as a special device for which students are charged $0.50 a page.

Backups. Although students should take some responsibility for backing up their work, there should be some straightforward facility to assist them. For example, a student should be able to upload an entire disk of materials to the campus utility for safekeeping.

Application-dependent support tools. Special projects may require special tools. Some projects should have access to a parser generator, lexical analyzer, or load generator. For example, we provided two such tools for our network class. The first was a network spy that could record all traffic on the network. Students used this tool to observe how their programs communicated over the network. The second was a packet injector, which could place an arbitrary kind of packet on the network. This allowed students to feed erroneous packets to their system for testing purposes.

Our list is not intended to be exhaustive, but we feel that the current state at most universities is far below even the suggestions above.

Using Macintoshes. Although the Macintosh might be familiar to the students and relatively inexpensive, we required substantial hardware and software facilities to use the Macintoshes in our laboratory. We were generally satisfied with the computational power of the system. The original Macintosh is a Motorola 68000-based system using between 128K bytes and 2M bytes of RAM. It contains a sound driver, an internal disk drive (approximately 400K bytes), a bit-mapped display (minimally configured with 512 × 342 pixels), and a mouse. A 64K-byte ROM provides some predefined operations for graphics, fonts, menus, windows, various device drivers, and other system utilities. Although useful, all the predefined operations may be replaced by routines in RAM or ignored by a program entirely. Full access to the underlying hardware is supported.

We also required a reasonable collection of network software and hardware for connecting Macintoshes together. We used materials from four sources: Apple, Kiewit (Dartmouth's Computation Center), Tanenbaum's text on computer networks,[8] and some materials we developed ourselves.

From Apple we acquired the basic AppleTalk software and documentation. The AppleTalk network architecture can be used to connect up to 32 hosts on a broadcast network. Multiple networks may be connected through bridges. A host uses a carrier-sense, multiple-access with collision-avoidance scheme to place packets on the network. The lowest level of the architecture uses an unreliable datagram protocol called LAP or link access protocol. The current implementation runs at 230.4K bits/s over a twisted pair of wires. The next higher layer, Datagram Delivery Protocol, or DDP, provides unreliable datagram delivery across multiple AppleTalk networks. A name binding protocol, a routing table maintenance protocol, a zone information protocol, and a transaction protocol all reside at the next higher layer. From Kiewit, we acquired the Kiewit Stream Protocol libraries that implement a reliable byte-stream protocol.[9] Because we were using a Pascal development system, we were able to implement directly several Pascal algorithms from Tanenbaum's book. Finally, we developed a set of libraries and programs that implemented a substantial fraction of DoD's Internet protocols (TCP/IP).[10,11]

When the course was run, our laboratory contained approximately 20 Macintoshes and four Lisas (Macintosh XLs) for use in the course—sometimes more Macintoshes and Lisas were available when students or faculty left their personal machines in the laboratory; sometimes fewer were available because of broken power supplies, mice, and keyboards. Students wrote programs in Lisa Pascal and Motorola 68000 assembler (as appropriate), and cross-compiled them on the Lisa for use on the Macintosh. Most Macintoshes were minimally configured, that is, had only 128K bytes of memory and no extra peripherals, but we had several machines with 512K bytes of memory, several external 400K-byte disk drives, and one 10M-byte internal disk. The Lisas could also be configured to act like Macintoshes with a megabyte of memory. We built cabling, terminators, and related hardware to support seven experimental networks that could be wired in many fashions. Students could produce hard copy output by attaching one of several Imagewriter dot-matrix printers to a Lisa or Macintosh, or by uploading the program (or bit-map image) to one of the time-shared computers and using a shared printer.

Students were permitted to use the equipment in any reasonable fashion. They were given 24 hour per day access to the materials. In addition, many students (and their roommates) owned Macintoshes that were connected to the campus AppleTalk network. Therefore they could use the equipment in their dormitories for some aspects of the course.

Our experiences

The Computer Network Laboratory class was designed to be a joint course between the Mathematics and Computer Science Department (COSC 88) and the Thayer School of Engineering (ENGG 128). The students taking this course would have completed prerequisites at the intermediate level in large system design and in computer architecture.

We offered the computer network laboratory as an elective during the 1985 winter quarter (about 10 weeks). We had 20 students, most of whom were junior or senior computer science majors. Most of the students had a reading knowledge of Pascal and had used the Macintosh for turn-key applications, such as word processing and figure drawing, but only one had any experience writing a stand-alone Macintosh program. We spent about two weeks teaching students how to write simple Macintosh applications and how to use the network software packages.

The students had to produce one program a week for the first six weeks. Most programs required the students to investigate some network feature. For example, a maze game illustrated a broadcast protocol, a sliding-window algorithm illustrated a byte-stream protocol, and a time-server program provided experience with soft layering of protocols. For each assignment, we provided a shell of a Macintosh source program for handling most of the operating system (Toolbox) overhead, a working sample solution (runnable application only) that could be run in a Macintosh, and a "broken" sample solution (runnable application only) that violated the protocol in some unspecified way. Our syllabus is reproduced in the sidebar A Network Course; a more detailed description of the laboratory assignments can be found in the sidebar Laboratory Assignments.

During the last four weeks of the term, the students worked on their own projects (usually two people to a project). Some of the projects were

- A 3-D maze game;
- A 2-D maze game allowing maze walls to be added or removed during play;
- An AppleTalk-Internet gateway with dynamic routing for load balancing;
- A multiconversation "CB" system that allows arbitrary subsets of participants to converse without interference from others (multiuser Unix Talk);
- Name and File Servers;
- Telnet and TCP packages; and
- DoD Internet gateway for moving IP packets between AppleTalk and a serial connection.

Some unique student experiences

For the most part, the projects were of the student's own design and construction. They applied the theories and techniques taught in class to construct a working network system. In doing so, they became aware of several issues to which they had no previous exposure. Three of these issues, consistency, security, and reliability, are discussed below.

Consistency. Because students were working with several versions of a program on more than one machine, they were faced with real problems of consistency. Not only did students have to ensure that each machine was running a working version of a program, but the program had to guard against some incompatible, older version of the program somehow remaining on a network. In contrast, a new release of a program on the central machine was installed over the old version and everyone used the new release. On the network, there was no way to recall old versions of programs. This was especially true for early releases of game programs that were distributed over the campus network to friends. Hence programs had to be written to recognize and take appropriate action when faced with inconsistent data generated by older versions of a program.

Security. A related problem concerned security. Many programs made assumptions that only well-behaved clients would use a protocol. Although gross errors in the protocol could be handled, a malicious program could cause inaccurate data to be introduced into many clients and the general strategy of a protocol subverted. A simple example came up in the context of game programs. As long as all programs accurately reported the actions taken by players, all players could be competitive. However, students soon learned how to write programs that could send data that would make a player appear invincible: players could move just before bullets came; scores could be adjusted arbitrarily during a consistency check; other players could be ignored. Indeed, these kinds of "malicious" programs were frequently designed along with the actual application as a way to monitor performance, test certain protocol features, and do other maintenance operations. Within our environment, the distribution of the maintenance program usually accompanied the distribution of the application; thus the security

A network course

Our Computer Network Laboratory course was designed as an elective course for students to gain experience with network protocols and distributed systems. Through experiments, we wanted students to observe the effects of unreliable communication, unstable network routing algorithms, window size on stream protocols, and malicious network agents. We also wanted students to construct broadcast-based applications, to use soft layering of protocols, to build a server, and to build a client. After students completed our laboratory exercises, we wanted them to design and implement a network project of their own choosing. We believe we met our objectives using the course format described below.

The course met twice a week. Each class meeting lasted 105 minutes with a 5- to 10-minute break between 50-minute halves. One class meeting a week was devoted to general network theory, while the other was devoted to a discussion of the week's laboratory assignment or a case study. Below we give a week-by-week description of the material that was covered in each lecture.

The readings for the course included documentation on the Macintosh's Toolbox,[1] documentation on AppleTalk protocols,[2,3] documentation on various DoD Internet protocols,[4-7] documentation on locally developed network protocols,[8] critiques of the ISO-OSI model[9] and protocol design in general,[10] overviews of Suns' NFS protocol,[11,12] and Tanenbaum's textbook on computer networks.[13]

Week	Theoretical part of the course
1	Overview of Networks: layering, ISO-OSI model description and evaluation
2	Network Design Issues: topology and its effect on performance, design of network topologies, review of graph theory
3	Physical Link Level: basic communications techniques
4	Data Link Level: flow control, error recovery
5	Network Level: network-wide control, queueing theory results
6	Routing, stability, multiple networks, bridges, gateways
7	Local area networks: contention, token passing
8	Transport Level: communications services for user processes, remote procedure calls, message passing operating systems, virtual sockets
9	Distributed computing systems; synchronization, two-phase commit, load sharing
10	Future of Computer Networks: technology trends, privacy, legal aspects, international networks

Week	Laboratory part of the course
1	Overview of Macintosh Operating System and Toolbox, Lisa Development System
2	Overview of AppleTalk architecture and implementation
3	Overview of DoD Internet Protocols, MacIP package, implementing soft layering of protocols
4	Broadcast and distributed protocols
5	Connection initiation, flow control, security issues
6	Design of gateways, bridges, addressing, connecting heterogeneous networks
7	Guest Lecture/Case Study: SUN's Network File System; Project ideas
8	Guest Lecture/Case Study: Apple's Printer Access Protocol
9	Final project review
10	Guest Lecture/Case Study: Practical issues of wiring dorms with AppleTalk, remote evaluation of network performance, maintenance problems, building codes.

References

1. Apple Computer Inc., *Inside Macintosh*, 20525 Mariani Ave., Cupertino, CA 95014, 1985.
2. Apple Computer Inc., *Applebus Developer's Handbook*, 20525 Mariani Ave., Cupertino, CA 95014, 1985.

of most systems built in class could be easily compromised.

Reliability. The students also had to deal with issues of reliability. Machines on the network, and even the network itself, could be disconnected or refuse to participate in a protocol. Programs had to validate information in a way that students never experienced: there was no guarantee that "reading" a remote location that had been "written" previously would yield the same value. Noise on the network could corrupt data; messages could be duplicated, arrive out of order, or never arrive. Trade-offs between tardy, reliable data and recent, unreliable data had to be accommodated. These issues of reliability were never explored on the time-sharing utility because the operating system took great care to make the user's interface look reliable. The ability to use a personal workstation allowed students to experience unreliability and experiment with ways to control it.

An optimistic future for laboratories in distributed systems

Our experiences with using workstations in a computer network laboratory were quite positive. We feel that the students learned a great deal from the environment; according to the class evaluation forms, the students agreed. Much to our surprise, the ratings given by the students for the value of the course and for the material learned were the highest that either of us have seen at Dartmouth. Perhaps the best estimator of our success was the number of anecdotes told to us by students who performed experiments of their own design that had no direct relation to any assignment—they just wanted to satisfy their own curiosity.

Our feelings and those of the students seem to match the expectations of instructors from other computer science departments: the use of workstations in this context was novel and exciting; few believed that we could run the course as we did. Colleagues at Brown, Cornell, Stanford, UCLA, and Carnegie-Mellon tell us of the many ways of using personal computers as replacements for their time-sharing systems, but few gave applications that let students exploit the ability to change the operating system, to alter the network capabilities, or to write distributed applications. From our conversations, we found an increasing use of bit-mapped graphics of workstations in courses. However, all the systems provided by the instructors were intended to be used as computer utilities, much like current computation center facilities. In contrast, the systems we wanted our students to build were to be highly experimental and usually unstable. We did not use workstations as replacements for central time-sharing computers or as advanced graphics terminals.

Our focus on letting students have full access to the facilities of a workstation also qualitatively changed the nature of the students' experiences. In general, students do not write privileged or distributed programs on computer utilities. Therefore they faced neither the technical issues of consistency, security, and reliability, nor the social and ethical implications of their systems. Because we encouraged our students to explore the interactions and performance of real systems, we ran into one controversy: protection of privacy vs. needs of maintenance programs. As we mentioned above, students learned the value of maintenance programs, and wrote them as necessary. Since the campus uses the same kind of network as we provided in our laboratory, the maintenance and monitor programs written by students for the experimental networks of the class function equally well on Dartmouth's public networks. The knowledge of how

(Sometimes referred to as *Inside AppleTalk.)*

3. Gursharan S. Sidhu and Alan B. Oppenheimer, *AppleTalk Printer Access Protocol,* tech. memo, Apple Computer 20525 Mariani Ave., Cupertino, CA 95014, Feb. 15, 1985. (Also published as part of *Inside LaserWriter,* which is available from Apple.)
4. Mark Sherman, *A Network Package for the Macintosh using the DoD Internet Protocols,* tech. memo, Dept. of Mathematics and Computer Science, Dartmouth College, Hanover, NH 03755, 1984.
5. Stanford Research Institute, *Internet Protocol Transition Workbook,* Network Information Center, SRI Int'l, Menlo Park, CA 94025, March 1982.
6. John L. Romkey, *IBM PC Network Programmer's Manual,* tech. memo, Computer Systems Research Group, Laboratory for Computer Science, MIT, Cambridge, MA 02139, 1984.
7. Samuel J. Leffler, William N. Joy, and Robert S. Fabry, *4.2BSD Networking Implementation Notes—Revised July, 1983,* tech. memo, Computer Systems Research Group, Computer Science Division, Dept. of Electrical Engineering and Computer Science, University of California, Berkeley, CA 94720, 1983.
8. Richard Brown, *The Kiewit Network Stream Protocol (KSP),* tech. memo, Kiewit Computation Center, Dartmouth College, Hanover, NH 03755, Dec. 18, 1984.
9. Danny Cohen and Jon Postel, *The ISO Reference Model and Other Protocol Architectures,* tech. report, Information Sciences Institute, University of Southern California, 4676 Admiralty Way, Marina del Rey, CA 90291, 1983.
10. J. H. Saltzer, D. P. Reed and D. D. Clark, "End-To-End Arguments in System Design," *ACM Trans. Computer Systems,* Vol. 2, No. 4, Nov. 1984, pp. 277-288.
11. Bob Lyon et al., *Overview of the Sun Network File System,* tech. memo, Sun Microsystems, 2550 Garcia Ave., Mountain View, CA 94043, Jan. 1985.
12. Anthony West, *The Sun Network File System, NFS—Business Overview,* tech. memo, Sun Microsystems, 2550 Garcia Ave., Mountain View, CA 94043.
13. Andrew S. Tanenbaum, *Computer Networks,* Prentice-Hall Inc., Englewood Cliffs, N.J., 1981.

to build and use these maintenance programs became a minor controversy on campus; the local student paper wrote editorials questioning the wisdom of allowing students to have monitoring programs that could be used on networks that carry private and possibly confidential information.[12,13] Although issues concerning privacy and ethics have always been included in computer science courses at Dartmouth, the issues now have a tangible feel that we hope students can appreciate better.

Distributed systems and networks are becoming an outstanding feature of the computer science landscape. Therefore we should be prepared to provide substantial laboratory materials to aid students in learning the principles, features, and limitations of this technology. We found the construction and use of a computer network laboratory to be feasible and effective. Macintoshes are sufficiently inexpensive that many can be purchased for use as a computer network laboratory, yet they are powerful and sophisticated enough to permit explorations of technical issues that cannot be pursued on central time-sharing computers or even workstations used as computer utilities. Because students can gain access to the most secure levels of the operating system in a workstation, they can affect the availability of resources and violate the privacy of fellow students. Thus students who participate in a course like ours are forced to consider the security and ethical implications of their work.

We are sufficiently encouraged by our use of workstations that we are preparing another course—this time in software engineering—that uses the capabilities of workstations. We already use the same equipment for our computer architecture course and the Physics Department offers a laboratory-equipment interface course using similar equipment with special controllers, amplifiers, and sensors. All three courses recognize the importance of distributed systems and networks, and each course discusses how that technology affects the subject. Through this article, we wish to encourage other universities to view workstations as a new, flexible resource that can be used to enrich courses well beyond a simple replacement of time-sharing machines for service courses. □

Laboratory assignments

The laboratory assignments were intended to provide experience with the design and performance of several network architectures. In particular, the students had to become comfortable with the available development facilities, the general organization of the communications software, and a variety of protocols.

Our plans originally included a hardware laboratory where students would experiment with a network interface and observe attenuation, echoes, and other phenomena. Unfortunately, we could not assemble the necessary facilities to provide this experiment. If we had had our hardware laboratory equipped before the course started, we would have assigned the experiment between the second and third laboratories listed below.

In addition to the hardware and software facilities of the Macintosh, students were also given accounts on a Unix machine, where they could use the mail system, print files, and backup disks.

For each assignment, the students were given a handout that described where to find appropriate documentation (essential given the sheer mass of documentation provided), a working sample solution (runnable object code only), source for the sample solution with the relevant parts of the assignment removed, and a broken sample solution (runnable object code only). We gave the partial source for the sample solution to minimize the busy work for students. Most programs for the Macintosh follow a standard form that initializes various facilities, dispatches events to appropriate routines, and maintains the visual coherence of the screen. We wanted the students to concentrate on the network issues and not on Macintosh details. Therefore we provided the sources for correct interactions with the operating system and toolbox, and let the students add the network facilities. For example, the provided source for the "talk" program in laboratory 5 created the two windows and provided procedures for writing and maintaining the text in the windows. The students wrote the procedures for timeouts, retransmitting packets, initial connection protocol, and so on.

We also provided broken programs, that is, programs that superficially seemed to work. However, the programs would violate the requirements of the assignment in various ways, for example, the broken program for laboratory 5 would send packets in the wrong order, duplicate packets, ignore received packets, and sporadically refuse acknowledgments. Students used these programs to aid their debugging, and we used the programs for evaluating the assignments. (As a practical matter, we combined both correct and broken solutions in one program that had a switch for controlling its mode of operation.)

We tried to give the student a reasonable debugging environment for each assignment. A typical debugging configuration used five minimally configured Macintoshes, as shown in Figure 1.

Three Macintoshes—the ones across the top of the figure—are connected to an AppleTalk network. Two of the Macintoshes run the student's application program, which communicates via whatever protocol is being studied. The figure shows the sample solution for laboratory 4, the Maze Game, running on the machines at the ends of the figure. Underneath each machine running the Maze Game is another Macintosh acting as a debugger. Debugging machines are connected to application machines by a serial link and control the Macintosh running the application program. A programmer may set breakpoints, examine and change memory, trace instructions, and perform the usual debugging operations on the application machine through a multiple-window, mouse-oriented program on the debugging machine.

Between the two application machines is a "spy"—a promiscuous listener on the network. It can display all network traffic in a packet format. The spy machine is used to observe how the two application machines are communicating. Although typical, the exact configuration in Figure 1 is not essential. Debugging machines may be removed, application machines may be added, multiple networks can be bridged (possibly using multiple spies), and a "packet injector" program can be run on a Macintosh connected to the network. These and other configurations were constructed by students during the course. Indeed, an essential element of the course requires reconfiguration of equipment for investigating different protocol issues.

The students were given 7-10 days for each laboratory listed below. Although our environment does not completely meet our set of criteria for development tools, our students did not complain about a lack of materials, an undue amount of busy work, or the total time to complete an assignment. We did receive several complaints that some assignments did not include references to documentation that were, in fact, required for completion of the assignment.

Assignments:

1. Practice using the Lisa to program the Macintosh: write a Pascal program that can create a window, write text to that

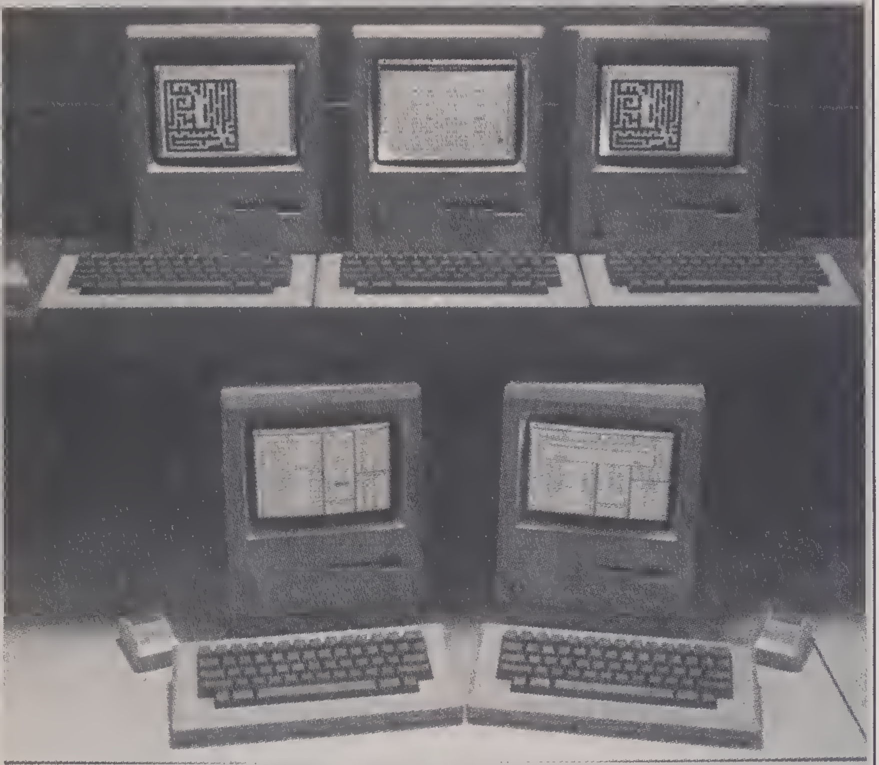


Figure 1. Five Macintoshes connected together serve as an experimental network for student assignments.

window, present dialogs to read strings and numbers, and manipulate various dialog controls (radio buttons, push buttons, check boxes); learn how to move files between Macintoshes, Lisas, and Unix.

2. Practice using the Macintosh Protocol Package and its tools: wire together several networks, write a Pascal program that can send and receive DDP packets, distinguish broadcast packets from single-destination packets, accommodate packets of many different types, maintain several socket connections, validate packets, inject packets into the network via the Poke program, and trace packets via the Peek program.

3. Write a Time Server and a Timer User: write a program that implements the DoD Internet time server protocol. The program should use UDP for packet delivery (UDP was provided). Multiple, simultaneous requests should be handled correctly. Write another program that asks the time from the server. The UDP package is implemented with soft layering, so the time server must be fit into the same paradigm.

4. Implement a distributed Maze game: write a program that uses a Maze-Game protocol for playing a version of Maze War. The protocol describes how players report their positions and interactions with other players. Students must also implement features of the protocol for players entering the game and leaving the game. No "master" host can be used—the protocol is egalitarian.

5. Implement a reliable byte stream: Use the P5 (full-duplex, sliding-window, byte-stream protocol with retransmission on time-out and piggy-backed acknowledgments) protocol given in Tanenbaum (P6 for extra credit—P6 uses a more sophisticated acknowledgment technique). Write a program that implements the Unix "talk" program on the Macintosh; that is, divide the screen into two windows where the left window displays all characters typed on a local machine and the right window displays all characters typed on a remote machine. The program should use AppleTalk's Name Binding Protocol for connection setup.

6. Measure file transfer protocols: Write a program that uses a half-duplex protocol (packet/acknowledge) for transferring a file using DDP. Compare the performance of the program with a (provided) program that implements the DoD Internet Trivial File Transfer Protocol (also a half-duplex protocol but with two extra protocol layers: UDP and IP).

Acknowledgments

Two students who took the Computer Network Laboratory course, Joel Margolese and Jeff Chase, reviewed this article and offered helpful suggestions.

Part of our equipment was provided by a grant to Dartmouth College by Apple Computer. Our IP packages were based on the PC IP implementation for the IBM PC developed at MIT[14,15]; our development of the Macintosh IP materials was supported in part by a grant from Apple Computer to Carnegie-Mellon University.[10] Other course development was supported in part by a grant from the Mellon Foundation and in part by a grant from the General Electric Foundation.

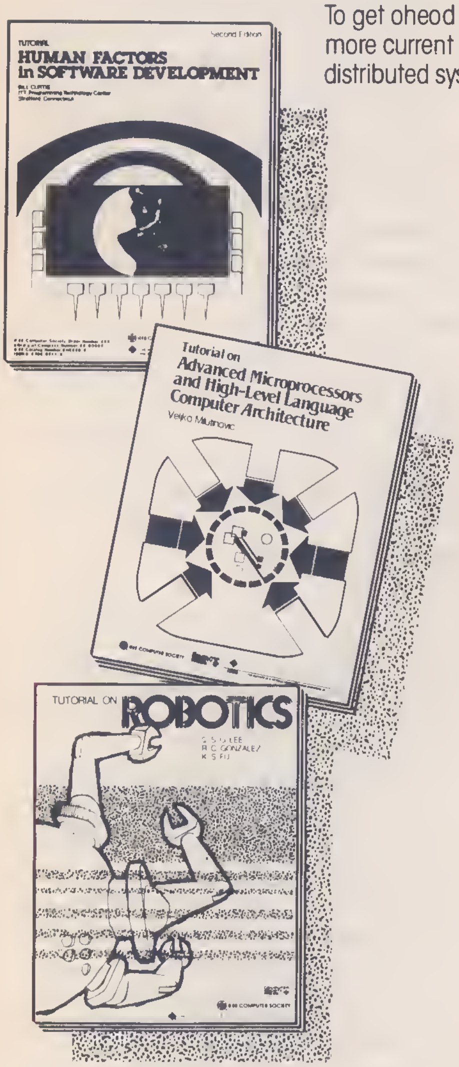
References

1. Marc H. Brown and Robert Sedgewick, "Progress Report: Brown University Instructional Computing Laboratory," *ACM SIGCSE 15th Annual Technical Symp. Computer Science Education,* Philadelphia, Penn., 1984.
2. Marc H. Brown and Robert Sedgewick, "A System for Algorithm Animation," *SIGGRAPH '84 Conf. Proc.,* Minneapolis, Minn., July 1984.
3. H. Kocak, M. Merzbacher, and M. Strickman, *Dynamical Systems with Computer Experiments at the Brown University Instructional Computing Laboratory,* tech. report CS-84-18, Dept. of Computer Science, Brown University, Providence, RI 02912, June 15, 1984.
4. Karen E. Smith and Elisabeth A. Waymire, *Brown University's Computerized Classroom: An Experiment in the Principles of Courseware Design,* tech. report CS-84-18, Dept. of Computer Science, Brown University, Providence, RI 02912, Oct. 2, 1984.
5. Edward Balkovich, Steven Lerman, and Richard P. Parmelee, "Computing in Higher Education: The Athena Experience," *CACM,* Vol. 28, No. 11, Nov. 1985, pp. 1214-1224.
6. Apple Computer Inc., *Inside Macintosh,* 20525 Mariani Ave., Cupertino, CA 95014, 1985.
7. Apple Computer Inc., *Applebus Developer's Handbook,* 20525 Mariani Ave., Cupertino, CA 95014, 1985. (Sometimes referred to as *Inside AppleTalk.*)
8. Andrew S. Tanenbaum, *Computer Networks,* Prentice-Hall Inc., Englewood Cliffs, N.J., 1981.

(*Continued on p. 41*)

the information edge

To get ahead and stay on top of your field, you need up-to-date information. No one gives you more current information than the IEEE Computer Society Press—software, computer networks, distributed systems, micros, and more—our titles give you an information edge in the information age.



JUST OUT!

Tutorial: Human Factors in Software Development (2nd Edition) *Bill Curtis*

The many new articles added to this latest edition will provide you with a greater understanding of, and appreciation for, the impact of cognitive science on software engineering. Contents: Cognitive Models of Programming Knowledge; Problem Solving and Design; Programming Language Characteristics; Fault Diagnosis; Future Directions in Programming; and more.
Order No. DB577, ISBN 0-8186-0577-4: March 1986, 736 pages
List price, $66.00/Member price, $33.00

Tutorial: Advanced Microprocessors and High-Level Language Computer Architectures *V. M. Milutinovic*

Provides computer system architects and designers, as well as computer users and managers with deep, broad, and up-to-date information on advanced topics in high-level language computer architecture. Contents: Reduced and Language-Directed Architectures; HLL Computer Architectures; Direct Execution Architectures; International Efforts; and more.
Order No. DB623, ISBN 0-8186-0623-1: January 1986, 608 pages
List price, $60.00/Member price, $36.00

Tutorial: Robotics (2nd Edition) *C.S. George Lee, R.C. Gonzalez, and K.S. Fu*

The latest edition of this popular tutorial gives an up to date summarization of the fundamental concepts and theories of robotics. Contents: Robot Arm Kinematics and Dynamics; Planning of Manipulator Trajectories; Robot Vision Systems; Robot Programming Languages; Machine Intelligence and Robot Planning; and more.
Order No. DB658, ISBN 0-8186-0658-4: April 1986, 744 pp.
List price $70.00/Member price $39.00

ALSO NEW:

Tutorial: Digital Image Processing and Analysis—Volume 1: Digital Image Processing *by Rama Chellappa and Alexander A. Sawchuk:* Order No. DB665, ISBN 0-8186-0665-7: June 1985, 736 pp.
List price $66.00/Member price $36.00

Tutorial: Digital Image Processing and Analysis—Volume 2: Digital Image Analysis *by Rama Chellappa and Alexander A. Sawchuk:* Order No. DB666, ISBN 0-8186-0666-5: Dec. 1985, 680 pp.
List price $66.00/Member price $36.00

Tutorial: Integrated Services Digital Networks (ISDN) *by William Stallings:* Order No. DB625, ISBN 0-8186-0625-8: Sept. 1985, 332 pp.
List price $39.00/Member price $27.00

Tutorial: Modern Design and Analysis of Discrete-Event Computer Simulations *by Edward J. Dudewicz and Zaven A. Karian:* Order No. DB597, ISBN 0-8186-0597-9: Oct. 1985, 480 pp.
List price $49.00/Member price $32.00

IEEE COMPUTER SOCIETY

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC
IEEE

COMPUTER SOCIETY PRESS

To order, return this form to: IEEE Computer Society Order Department P.O. Box 80452 Worldway Postal Center Los Angeles, CA 90080 U.S.A.
Visa, MasterCard, American Express or checks accepted.
To order by phone call toll-free: 1-800-272-6657 or 1-800-CS-Books (California residents call 714/821-8380), 8:00 am to 4:30 Pacific time.
For further assistance call 714/821-8380.

Indicate method of payment (orders must be prepaid)
☐ Check enclosed
☐ Visa ☐ MasterCard ☐ American Express
(Note minimum credit card charge is $25.00)

Card no. Exp. date

Signature

Name
(please print or type)
Address

City State Zip Country

Phone/Telex No. IEEE/Computer Society Member No.
(Required for member discount)

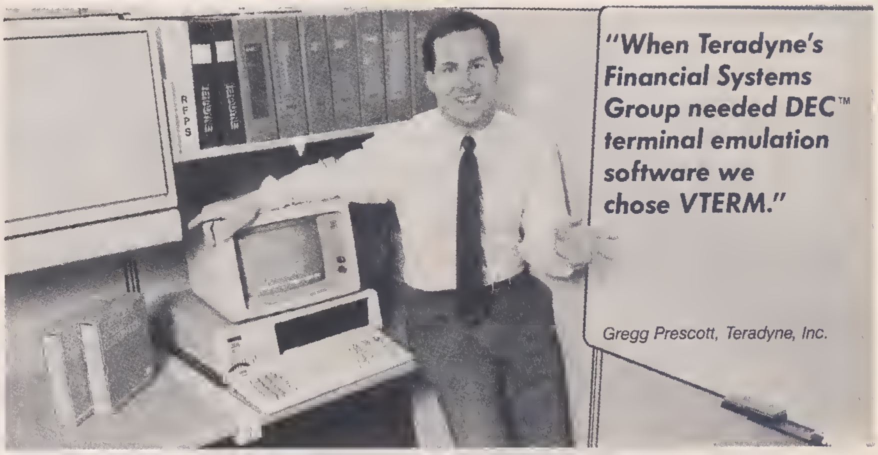
Quantity	Order No.	Title	Price	Total
	DB			
	DB			
	DB			

Sub Total ______

California residents add 6% sales tax ______

TOTAL Enclosed ______

9. Richard Brown, *The Kiewit Network Stream Protocol (KSP)*, tech. memo, Kiewit Computation Center, Dartmouth College, Hanover, NH 03755, Dec. 18, 1984.
10. Mark Sherman, *A Network Package for the Macintosh using the DoD Internet Protocols*, tech. memo, Dept. of Mathematics and Computer Science, Dartmouth College, Hanover, NH 03755, 1984.
11. Stanford Research Institute, *Internet Protocol Transition Workbook*, Network Information Center, SRI Int'l, Menlo Park, CA 94025, March 1982.
12. Esther Schrader, ‘‘‘Peek disk’ raises Computer Science ethics doubts: used for ‘Networking’,’’ *The Dartmouth*, HB 6175, Dartmouth College, Hanover, NH 03755, Jan. 8, 1985, pp. 1-2.
13. Editorial, *The Dartmouth*, HB 6175, Dartmouth College, Hanover, NH 03755, Feb. 12, 1985, p. 10.
14. John L. Romkey, *IBM PC Network Programmer's Manual*, tech. memo, Computer Systems Research Group, Laboratory for Computer Science, MIT, Cambridge, MA 02139, 1984.
15. Jerome H. Saltzer et al., "The Desktop Computer as a Network Participant," *IEEE J. Selected Areas in Communication*, Vol. SAC-3, No. 3, May 1985, pp. 468-478.


"When Teradyne's Financial Systems Group needed DEC™ terminal emulation software we chose VTERM."

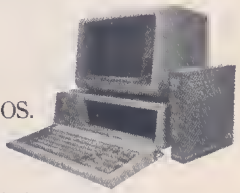
Gregg Prescott, Teradyne, Inc.

VTERM/220 Quality makes all other DEC terminal emulators obsolete

Over 35,000 demanding professionals, like Teradyne's Gregg Prescott, have recognized VTERM's superior quality. Now this same VTERM quality is available in Coefficient's powerful new emulator, VTERM/220.

Features include:

- Plug compatible VT220 emulation with 132-column support and optional Tektronix™ 4010/4014 graphics.
- Extensive file transfer system offering KERMIT, XMODEM and our VTRANS protocol with VMS,™ RSX11 M/M+,™ RSTS/E™ and UNIX™ software.
- Host data capture on PC with conversion to Lotus® 1-2-3,® Symphony® and dBase.®
- "Hot Key" toggle between host session and PC DOS.



Call us today at 212-777-6707 ext. 153.

Coefficient
The Leader in DEC Emulation Software
Coefficient Systems Corporation
611 Broadway, New York, N.Y. 10012

Trademarks: DEC, VMS, RSTS/E, RSX11 M/M+, Digital Equipment Corp.; Tektronix, Tektronix, Inc.; Lotus, 1-2-3, Symphony, Lotus Development Corp.; dBase, Ashton-Tate, UNIX, AT&T, Bell Laboratories.

Reader Service Number 2

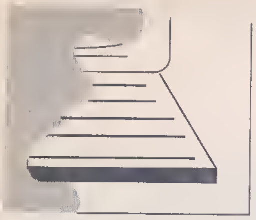
Mark Sherman is an assistant professor of computer science at Dartmouth College. He has worked previously at IBM Research, Lawrence Livermore National Laboratories, and Intermetrics. He has conducted research in a variety of areas including programming language design and implementation, computer networks and software engineering. His current research is focused on graphical programming languages.

Sherman received his undergraduate degrees from MIT in 1977 and his PhD degree in computer science from Carnegie-Mellon University in 1983. He is a member of Sigma Xi, Eta Kappa Nu, and Tau Beta Pi.

Ann Marks is an assistant professor of engineering at Dartmouth College (Thayer School of Engineering). Marks has worked previously at IBM Federal Systems Division and Sanders. Her current research interests are in distributed operating systems and reliable system design.

Marks received her BS (1976), MEng (1977), and PhD (1980) degrees in electrical engineering from Cornell University.

Readers may write to Sherman at Dartmouth College, Dept. of Mathematics and Computer Science, Bradley Hall, Hanover, NH 03755.



Vocationalism and the Whole Man

Gerald L. Engel, University of Connecticut
John F. Dalphin, Norwich University

High-quality computer science programs may be accredited as long as they meet the spirit of the CSAB criteria.

The establishment of the Computing Sciences Accreditation Board, or CSAB,* and the development and implementation of an accreditation program in computer science has caused concern among individuals involved with liberal arts programs. Many who are associated with computer science programs housed in a liberal arts structure believe that their programs are of high quality and able to produce graduates competitive with those of more technically oriented curricula; yet they are concerned that their programs may not meet the letter of the CSAB criteria.[1,2]

Traditionally, liberal arts schools and colleges have been concerned with accreditation at only the institutional level, that of their regional accrediting association. Program accreditation has been perceived as an exercise for more narrowly focused professionally oriented programs, while the liberal arts curriculum was aimed at educating the "whole person." The breadth of the liberal arts program and its value in preparing students for future development have been thought to benefit the graduate in the long term in contrast with professional programs, which are often perceived to prepare a more technically job-ready graduate.

In recent years liberal arts schools and colleges have been reevaluating their position. As students have become more goal oriented and look for programs that will allow direct entry into the workplace, programs with strong professional orientation in education, business, and now computer science have been added to the traditional liberal arts structure.

Computer science programs highlight the conflict of professionalism with the traditional role of the liberal arts. With its intellectual base, its foundation in mathematics and the sciences, and its pervasive application across all disciplines, computer science qualifies to be considered one of the "liberal arts and sciences." Yet there is a professional aspect to the discipline in that many, if not most, individuals completing a baccalaureate in computer science immediately seek employment in the computer field.

Accreditation of baccalaureate-level programs in computer science was begun in part to help ensure that those individuals moving into the computer profession after completing their degrees would be reasonably qualified to do so. The criteria established by CSAB to review programs

---

*CSAB was incorporated in the state of New York in 1985. The current members of CSAB are the IEEE Computer Society and the Association for Computing Machinery.

0018-9162/86/0600-0042$01.00 © 1986 IEEE

seeking accreditation have been viewed by many at liberal arts schools and colleges as being too engineering or professionally oriented and unable to evaluate a high quality liberal arts program. Most often mentioned as specific concerns are the following:

- The "approximately one-and-one-half years of computer science coursework" of the criteria in contrast to institutional limitations on the number of courses that may be taken in one field or area to satisfy requirements for a major,
- The minimum number of faculty (five) to teach the program, and
- The possible public perception that a program is lacking in quality if the institution elects not to seek its accreditation.

To address these issues, the Educational Activities Board, or EAB, of the IEEE Computer Society, or IEEE/CS, and the Education Board of the Association for Computing Machinery, or ACM, funded two workshops held in Chicago May 31-June 2 and October 11-13, 1985. The workshops were designed to address the above and related problems and to make recommendations to the CSAB regarding computer science programs housed in liberal arts schools and colleges. A list of participants is in the appendix.

Although many computer science programs that are in a liberal arts structure are taught at small colleges, it is important to note that the charge to those participating in the workshops did not include considering the general question of computer science at the small college. It specifically was not the charge of these workshops to address issues pertinent only to small colleges, nor to address programs that—while having significant computer study—are not computer science programs. Thus this article does not consider programs in information systems, data processing, or a variety of like areas that may involve joint minors, or their equivalent, in mathematics and computer science. This is not meant to impugn the quality of such programs; rather they were beyond the scope of these workshops.

Concerns with the CSAB critiera

The most frequently voiced concern with the CSAB Computer Science Accreditation Criteria is that it is "too engineering oriented." This usually translates to concerns over the curriculum requirements, and the faculty requirements. The specific curriculum requirements that seem to pose the greatest difficulty are the computer science requirements (1.5 years, or 37.5 percent of the four-year program), the mathematics and science requirements (1 year, or 25 percent of the program), and the corresponding low percentage of general education requirements and elective opportunities. The faculty concerns are with the requirement for five full-time-equivalent faculty (four full-time with primary commitment to the program) and some aspects of the teaching load such as the normal maximum of two course preparations per term.

Many liberal arts schools and colleges restrict the number of hours that may be required in a program or major; this often prevents their computer science program from directly meeting the curriculum criteria. There does not appear to be any uniform approach to these restrictions. One institution may allow no more than 25 percent of a program to be in a major, but in some cases introductory courses are not counted. Another institution places restrictions on the maximum number of required courses regardless of the department in which they reside.

One extreme situation discussed at the workshops was a program housed in a combined department of mathematics and computer science. Institutional regulations placed course limits on majors by department; thus there was no hope of the program meeting the computer science and mathematics requirements of the CSAB criteria. Yet if there were two departments for mathematics and computer science, the CSAB criteria of approximately 37.5 percent computer science and approximately 12.5 percent mathematics could be met within institutional regulations.

The variations existent around the country seem endless. It appears that any set of curriculum criteria that would serve one group of schools would not serve some other group.

The concerns about number of faculty and faculty loads are primarily a function of the size of the school and the number of students enrolled. Departments with relatively small numbers of majors often cannot justify five full-time-equivalent faculty, and breadth in the curriculum often requires more than two preparation.

While the issues of faculty load, and size, seem solvable (indeed, upward adjustments in a program's curriculum requirements may increase the need for faculty), the curriculum area is more difficult. It has been noted that no single set of curriculum criteria will meet all liberal arts program definitions, nor does it seem that the liberal arts experience can be quantified for measurement. However, the attendees at the workshops believe that there are factors that can be identified to compensate for apparent deficiencies when the CSAB criteria are applied strictly. Some of the common themes in this regard are small class size, a sense of institutional identity, low student to faculty ratios, individual attention, and easy access to equipment.

One of the major efforts of the workshops became the identification of these compensating factors.

Approaches to the concerns

In the two workshops we approached the concerns in the following five ways:

1. We considered specifications of a computer science (and some others such as science and mathematics) core component of "quality" programs.
2. We considered a statement that accreditation is inappropriate for programs in the liberal arts.
3. We considered new sets of criteria specifically directed to programs in the liberal arts.
4. We considered revising the existing criteria to broaden them.
5. We analyzed the existing criteria to see if they are sufficent and can be applied in the liberal arts context through recognition of balancing factors.

Each will be discussed in detail.

Specifications. The idea behind attempting to specify a computer science core was to provide an alternative to the general computer science categories outlined in CSAB's criteria. After some discussion it was noted that although such an exercise could be useful at some point in

Many programs may be computer related rather than "computer science" and not fall under the CSAB criteria.

time, it might result in a superficial listing of the topics taught in programs.

It also became evident that the expression of criteria in a general framework (as CSAB has done) was more appropriate. If model curricula are to be developed, the conclusion of the workshops was that the sponsoring societies should be responsible for that effort and should promulgate the more detailed recommendations.

Inappropriateness of accreditation. That accreditation is inappropriate for liberal arts colleges was argued eloquently by one of the participants at the first workshop. It was suggested that the following addenda be incorporated in the published CSAB criteria:

> We recognize that many BA granting colleges and universities have developed (or plan to develop) computer science major programs with different overall aims and objectives than those to which this accreditation standard addresses itself. Such institutions prefer to maintain high standards of quality through other avenues of accreditation and for the BA degree in its entirety rather than for separate major fields individually. For such institutions, we acknowledge that the computer science accreditation is not appropriate.

There are a number of cases for which this is true. In some programs the ultimate objective is to prepare the student for additional specialized study following the baccalaureate degree. Where this is the case and the graduates move directly into graduate study, there is less need for the undergraduate program to seek accreditation since depth in computer science would be achieved at the graduate level.

Also there are many institutions where the graduates are actively sought and employed by industry upon graduation. If these employers are pleased with the graduates, and the institution does not intend to expand beyond this role, there is little reason to think that the opinion of the employers will change simply because the institution does not seek accreditation.

Finally, many programs may be computer related rather than "computer science" and not fall under the CSAB criteria.

While the workshop attendees recognized that it might not be appropriate for all institutions to seek accreditation, they took a more positive approach. With the exceptions noted, there are in fact few institutions that can ignore a widely accepted accreditation program. Also, few people looking at programs from outside academe can distinguish between a BA and a BS program (or differentiate between a professional program and a nonprofessional program). Thus, if accreditation in the computing sciences becomes generally accepted, the question will be whether or not any specific program is accredited. For some institutions to remain competitive, accreditation may be a necessity.

Criteria for the liberal arts. The preparation of a set of criteria targeted at programs housed in a liberal arts structure was considered only briefly. While this was recognized as appropriate for programs with distinctly different emphases (such as those usually called information systems programs), workshop attendees were concerned that such an approach might create one set of criteria for "real" programs and one set for "second-class" programs. The participants did not want to risk this happening.

Rewriting existing criteria. Reworking the existing criteria to broaden them was given considerable time and effort during the two workshops, particularly the June meeting. One "strawman" considered in depth involved a computer science requirement of one year with the mathematics and science supporting disciplines at three-quarters of a year.

Although this would have allowed several of the institutions represented to meet the criteria, not all would. In addition, it appeared to the attendees that there was a greater possibility that some institutions with poor quality programs might meet the criteria. The exercise resulted in one participant remarking that he "became impressed with the work that went into crafting the CSAB criteria."

The consensus was that the existing CSAB criteria provided a defensible line to not accredit programs. Lowering the requirements to the "strawman" level created a line too low; hence it would be a mistake to "tamper" with the criteria in order to bring in one class of programs.

By the time of the second workshop, participants had rcconsidered the issue with the perspective of time. After detailed discussion of the existing criteria, three items were put forward to be considered by CSAB for change:

1. In the intent statement of the criteria, specifically indicate that they are intended for programs in liberal arts and are flexible enough for such quality programs to be accredited.
2. Modify the curriculum section to require "1-¼ to 1-½ years of computer science."
3. Modify the science section to allow courses above the two-course laboratory science sequence to be in science or empirical methods.

The first proposed change is needed to encourage liberal arts administrators to consider seeking accreditation. The second provides a range of coursework (as opposed to the "approximately 1-½ year" existing criterion). The participants indicated that in a high-quality environment, 1-¼ years of computer science should be sufficient. If this range were to be adopted, the participants saw little that would allow a program to go below 1-¼ years of computer science. The third proposed change was thought to provide greater flexibility.

These proposed changes were perceived as making accreditation accessible to more institutions. Even with these proposed changes, some of the programs at institutions represented at the workshops still would not meet the criteria.

Analysis of existing criteria. Recognizing that the proposed modifications might not be accepted by CSAB (and even if they were, there still would be a need for interpretation of whatever criteria exist), the participants developed some intepretations that could accompany the CSAB criteria. The concept behind these proposed interpretations is to specify those factors that could provide experiences equivalent to those arrived at by meeting the letter of the CSAB criteria. Some of these proposed "balancing" factors are:

1. Limited course offerings might be balanced by such activities as independent study, thesis, senior projects, required seminars, cooperative education, and work-study experiences.

2. Where there are smaller class sizes and well-prepared students, a smaller offering of courses in the curriculum may be balanced by greater richness of topics covered in the courses.

3. Greater student contact with faculty (lower student to faculty ratios) may compensate for curriculum content or minimum faculty requirements.

4. A program's general education requirements may relate directly to some aspects of the computer science component of the curriculum. Such relevance and integrations could be a compensating factor with respect to the minimum computer science requirement.

5. The care with which students are advised may be a balancing factor in evaluating the depth and breadth of the computer science content of the program.

6. Greater access to laboratory and computing resources than that outlined in the CSAB criteria may provide greater depth in the students' computer science studies.

7. Institutional strengths (such as strong admissions criteria, success of graduates, and overall rankings in serious evaluations) may compensate for weaknesses in other areas.

In order to take advantage of such interpretations, it would be necessary that the institution develop and supply the significant supporting evidence. For example, an institution with a strong liberal arts philosophy should be certain to express this in a clear statement that shows how computer science study is enhanced by this focus. At the same time, it would be necessary to ensure that CSAB visiting team members be sensitized to the issues involved.

In reviewing the compensating factors identified, the workshop participants recognized that they should not be considered unique to liberal arts-oriented programs, but rather could help in identifying quality programs in any environment. Furthermore, it was recognized that with greater study a larger list of compensating factors might be identified. The experience of actual visits may provide some items for consideration.

The workshop attendees believe that it is important that an interpretive document (or documents reflecting varieties of settings) with as wide a focus as possible be developed soon. While it appears that such a task is more appropriate for the sponsoring societies, it is important that the experiences of the current CSAB visits be factored into the development.

Summary recommendations

The following recommendations were adopted by the participants in the two workshops;

1. The following proposed modifications to the accreditation criteria be conveyed to the Computing Sciences Accreditation Board:

    a. Wording in the criteria be included to encourage institutions with programs in the liberal arts to seek accreditation;

    b. The requirement of "approximately 1½ years of computer science" be changed to "1¼ to 1½ years of computer science"; and

    c. The requirement for two science courses beyond the two-course sequence in the laboratory sciences be broadened to include empirical methods.

2. The nature of balancing factors that may be present in quality computer science programs be communicated to the Computing Sciences Accreditation Board. Furthermore, that examples of such factors be incorporated in the training sessions for team chairs and team visitors;

3. Sponsoring societies be informed of the desire for the development of interpretive guidelines and model programs that will supplement and expand upon CSAB criteria;

4. The accreditation visits and actions be evaluated—particularly those involving programs in a liberal arts structure—to determine what problems may exist, if any, for such programs being considered under existing criteria; and

5. The general endorsement of the concept of accreditation for computer science programs in the liberal arts context, as the sense of the workshops, be made known.

Modification to the CSAB criteria are possible that allow certain quality liberal arts programs to fit the letter of the criteria.

Attempting to draw conclusions from the deliberations of such a varied group of individuals representing a wide variety of programs can be dangerous. There were, however, several common threads stemming from the discussion at the workshops that warrant comment.

Perhaps the strongest result was recognizing that the liberal arts programs and their environments are so diverse that a single set of recommended criteria to cover them all is impossible. A corollary to this is recognizing that quality programs are in no way the sole property of liberal arts schools and colleges. Indeed, a variety of characteristics exist within high-quality programs, and any accreditation activity must seek to enhance and promote such quality wherever it may be found.

Modifications to the CSAB criteria are possible that allow certain quality liberal arts programs to fit the letter of the criteria. Where it can be shown that these modifications are beneficial to the process—and they will not adversely affect the quality of the accreditation effort—they should be incorporated into the criteria. It must continue to be the concern of CSAB to review and update its criteria to respond to the issues raised in the two workshops and to meet the ever-evolving needs of the computing discipline and profession.

Equally important is the development of interpretive guidelines for the criteria.

While the criteria are designed to be applied flexibly, appropriate papers and reports that will explain what this means to institutions—and to CSAB visitors—will go a long way toward providing a stronger accreditation program (without the complications of changing the criteria). The preparation and dissemination of interpretive documents must be a continuing high-priority activity of the sponsoring societies.

While it is recognized that not all liberal arts-based programs will seek accreditation, and while it is acknowleged that there are cases where accreditation may not be appropriate, it is believed that CSAB accreditation is appropriate for a great number of such programs. This accreditation should be regarded as an opportunity to measure a program's quality, to upgrade that quality where needed, and to provide recognition of programs of quality.

One key to the success of the process will be the number of institutions with strong programs that seek to have them accredited. Equally important will be the recruitment of professionals of recognized excellence to serve as team chairs and visitors; their contribution of time and effort will help ensure the success of the accreditation program under CSAB. Meeting these goals will help establish a quality accreditation program and contribute to the further development of the disciplines we know as the computing sciences. ☐

Acknowledgments

We want to acknowledge the assistance of all the workshop participants for providing the ideas contained in this article, especially J. Michael Dunlap, who provided a summary of the second workshop session. We would also like to acknowledge the assistance of the Educational Activities Board of the IEEE Computer Society, J. Thomas Cain, Chairperson, and the Education Board of the Association for Computing Machinery, Robert Aiken, Chairperson, for sponsoring this series of workshops. Finally, we want to acknowledge the assistance of Gerry Segal of the Association for Computing Machinery for taking care of the arrangements for the workshops.

References

1. Michael C. Mulder and John Dalphin, "Computer Science Program Requirements and Accreditation," *CACM*, Vol. 27, No. 4, April 1984; *Computer*, Vol. 17, No. 4, April 1984.
2. CSAB, *Criteria for Accrediting Programs in Computer Science in the United States*, Sept. 25, 1985, Computing Sciences Accreditation Board, 345 East 47th Street, New York, NY 10017.

Appendix

1. Attendees of the May 31-June 2, 1985 workshop:
   Della Bonnette, University of Southwestern Louisiana
   Robert Cannon, University of South Carolina
   Robert D. Cupper, Allegheny College
   John Dalphin, Norwich University, Cochair
   J. Michael Dunlap, Willamette University
   Gerald L. Engel, University of Connecticut, Cochair
   Saj-Nicole A. Joni, Wellesley College
   Michael Schneider, Macalester College
   Greg W. Scragg, Williams College
   Gerry Segal, Association for Computing Machinery
   Ted Sjoerdsma, Washington and Lee University
   Allen B. Tucker, Colgate University
   A. J. Turner, Clemson University
2. Attendees of the October 11-13, 1985 workshop:
   Della Bonnette, University of Southwestern Louisiana
   Robert Cannon, University of South Carolina
   Robert D. Cupper, Allegheny College
   John F. Dalphin, Norwich University, Cochair
   J. Michael Dunlap, Willamette University
   Gerald L. Engel, University of Connecticut, Cochair
   Greg Scragg, Williams College
   Ted Sjoerdsma, Washington and Lee University
   A. J. Turner, Clemson University
   Stu Zweben, Ohio State University

Gerald L. Engel occupies the Stamford Chair of Computer Science and Engineering at the University of Connecticut, Stamford. Active in the field of computer education for more than 15 years, Engel has chaired the steering committee of the National Educational Computing Conference and has served as its general chairperson and coprogram chairperson. Additionally, he was conference chairperson for the 1985 World Conference on Computers in Education.

Involved with the development of the first model program in computer science and engineering of the IEEE Computer Society, he has also served as vice chairperson of the Educational Activities Board of the society, and the Education Board of the ACM. He is currently the chairperson of the Education Committee of the American Federation of Information Processing Societies.

Engel received the BS degree from Hampden-Sydney College, the MA from Louisiana State University, and the DEd in Computer Science from Pennsylvania State University.

Readers may write to Engel at the University of Connecticut, Scofieldtown Road, Stamford, CT 06903.

John F. Dalphin is a professor of computer science and Special Assistant to the President at Norwich University in Vermont; he has also served as Vice President for Academic Affairs and Dean of the Faculty at Norwich. He is a member of the board of directors of the Computing Sciences Accreditation Board, Inc. and of the board of directors of NERCOMP. He is chairman of the Accreditation Committee and a member of the Education Board of the ACM.

Dalphin earned a BME at Clarkson College in 1962, an MS in mathematics at the University of New Hampshire in 1964, and a PhD in mathematics at Clarkson in 1973.

Computing Sciences Accreditation Pro and Con

Ware Myers, Contributing Editor

Society's need for generalists and specialists creates conflict in curricula development.

Is computer science a profession? was asked by J. Dennis Bjornson of the Boeing Company at the March 6 Compcon Spring panel session, which was moderated by J. T. (Tom) Cain of the University of Pittsburgh. Not legally, Bjornson answered.

That will take an accrediting process.

Such a process, operated by the Computing Sciences Accreditation Board, commenced operations in 1985, reported Taylor L. Booth of the University of Connecticut. Booth is the past president of CSAB and currently a representative director to CSAB for the IEEE Computer Society.

The development of the CSAB accreditation program "has caused concern among individuals involved in liberal arts programs," noted Gerald L. Engel, also of the University of Connecticut, a visiting team chair for CSAB.

"I am not generally a fan of accreditation," said Gene Lawler of the College of Engineering of the University of California, Berkeley, living up to Engel's observation. Lawler, a professor of electrical engineering and computer science, administers the computer science major for the College of Letters and Sciences.

Yale Patt, also of the College of Engineering of UCB but a proponent of accreditation, pointed out, "The issue is not making accreditation standards fit what the universities can satisfy. Rather the issue has got to be making accreditation standards fit what is needed by society."

Profession defined. Under the Fair Labor Standards Act of 1938 employees must be paid time and one half their regular hourly rate for hours over 40 per week. Executive, administrative, and professional personnel are exempt from this requirement, but they must meet certain criteria. In the case of the professional exemption, which might apply to computer science personnel, the criterion of interest here is the long and organized course of study on which professional status is based.

"The Boeing Company classifies those computer scientists who are a part of our Engineering team as part of our exempt professional engineering payroll," Bjornson said. "The typical work performed by these computer scientists is requirements definition, preliminary and detail design, programming, and integration and testing of typically large-scale, dynamic systems,

0018-9162/86/0600-0047$01.00 © 1986 IEEE

which include the hardware and software that are an integral part of our products.

"In regard to computer programmers and system analysts being recognized as learned professionals, however, the Fair Labor Standards Act is quite explicit," he continued. "Apparently, because of the lack of consistency and uniformity in computer science curricula, computer science is not recognized as a learned profession," Bjornson said. He quoted the Code of Federal Regulations, which implements the Fair Labor Standards Act:

"At the present time [1984] there is too great a variation in standards and academic requirements to conclude that employees employed in such occupations are a part of a true profession recognized as such by the academic community with universally accepted standards for employment in the field."

As an interim solution, Boeing devised a profile of minimum requirements and matched these requirements against the characteristics of several hundred existing computer science programs. Boeing then evaluated the programs on its provisional list and determined that just under half of them were qualified as sources of professional candidates.

However, Boeing would prefer to rely on a formal accreditation process operated by the practicing professionals in the field, as it does in the case of other disciplines. For example, it hires only graduates of programs sanctioned by the Accreditation Board for Engineering and Technology into positions under the professional exemption. It intends to use programs accredited by CSAB in the same way.

"Accreditation according to professionally established guidelines is the only workable solution to the computer science personnel dilemma," Bjornson concluded.

Wide range of programs. Nationwide there are about 1200 computer science programs offering various degrees. "They cover an extremely wide range of courses," Booth said.

One school believed its program to be outstanding—it had graduates at all the big companies who were doing well, it told the Board. Its program consisted of a required course in business programming—Cobol, according to Booth; a required course in scientific programming—Fortran; a course in database processing "which I think is best classified as dBase 3"; a course in the fundamentals of operating systems, "which was how to use MS DOS"; and a work experience period, which it called internship.

"And they felt this was preparing people for work in the computer field," Booth exclaimed in disbelief.

In another situation CSAB was contacted by an attorney on behalf of a client who had graduated from a computer science program, but had been dismissed after six months on the basis of incompetence. It turned out that the client had no more than high school mathematics and had taken only three courses in programming—Cobol, Fortran, and a report generator. Three such courses are not sufficient to make a valid computer science curriculum, Booth said.

"Many people are anxious to get into the computer field, but they have no idea of what the course of study should include," Booth went on. "They go to the local campus of a higher education institution and run into experiences such as these."

CSAB criteria. The key CSAB criteria include approximately 1.5 years, or 37.5 percent, of the four-year program in computer science and approximately one year, or 25 percent, in mathematics and science. In addition to courses in the humanities, social sciences, and other disciplines, the program should be rounded out by a selection of electives that would provide an understanding of an area to which the techniques of the computer field may be applied.

There should be at least five full-time-equivalent faculty of which four should be full time with primary commitment to the program. In addition, the criteria set standards for teaching loads, class size, laboratory and computing resources, and administrative support. (See J. T. Cain's article on CSAB in the January 1986 issue of *Computer* for further information.)

During its first year, CSAB made accreditation visits to 31 colleges and universities of the 125 that had requested visits, Booth reported. Final decisions will be made by the Board this summer. In the coming year CSAB plans to visit about 40 departments.

Two workshops held. The concern with the CSAB criteria by the liberal arts community led to two workshops in 1985 to address the accreditation issue. The complaint most commonly expressed was that the CSAB criteria are too engineering-oriented, reported Engel, the workshop co-chair. The nub of the issue seems to be that many liberal arts colleges limit the hours that may be devoted to the major to no more than, for example, 25 percent. Other institutions place a limit on the number of courses that may be required in the major. The intent of these limits is to ensure that students are broadly educated in the humanities, not just in a special field.

A report on the two workshops by Engel and John F. Dalphin is contained in this issue of *Computer.*

The liberal arts view. According to Lawler's interpretation of the CSAB criteria, 75 semester hours out of 120 are prescribed for computer science, mathematics, and science. In addition, 15 more hours of electives are supposed to be "computer-related." That makes 90 hours in total. At UC Berkeley, the computer science major for liberal arts students devotes 57 hours to these subjects. It devotes 33 hours to computer science itself, far less than the 45 hours specified by the CSAB criteria. In comparison, the ACM model curriculum requires only 33 to 36 hours in this area, Lawler said. (Berkeley also has a computer science program for engineering students that Lawler did not speak about.)

"The end result is that we are light years away from satisfying the CSAB requirements in the liberal arts program at Berkeley," Lawler asserted. "I think that it is exceedingly unlikely that we would like to have CSAB accredit our letters and science program in computer science. Yet I think it is a very high quality program and I don't think the faculty would want to change it to the extent required to meet the guidelines. There is absolutely no disgrace in a liberal arts program not being accredited by CSAB."

Quite a number of schools among those 1200 colleges with computer science programs of some sort will feel that, in order to achieve some kind of credibility, they must knuckle under to the pressure for accreditation and greatly expand their computer science requirement, Lawler continued. The result will be that "resources will be stretched still thinner and students will be given far less exposure to other subject matter that one would normally expect within a liberal arts curriculum," Lawler added. "It's just one more body blow to liberal education."

In his opinion, the CSAB thrust is to provide a standard for professional or engineering accreditation, a goal with which undergraduate liberal arts programs should not be concerned. When a student majors in physics, for example, it is not anticipated that he or she will become a professional until he gets a master's degree. "I think the same thing should be true in computer science," he indicated. "To become a professional, one should go on to graduate school."

Society's needs. There should be no loss of face for a liberal arts program to elect not to be accredited, because accreditation is really a form of licensing, Patt offered. By its very nature computer science education is double-headed on many levels. It is driven by economics in some ways, but not in all ways. There are programs with a strong intellectual content, such as the one at Berkeley, a content independent of the jobs in Silicon Valley, but there are other programs driven by jobs.

On the other side of computer science —the nonintellectual content part of it—it is a discipline that serves the needs of society, Patt observed. For that there has to be some form of licensing. In the absence of a CPA or bar exam, accreditation provides a form of licensing.

Patt saw two aspects in the accreditation process. One is the quality of the program and the other is its scope. A program that offers first class work in a very narrow aspect of computer science should be allowed to give computer science degrees, probably the AB. It shouldn't necessarily be accredited.

At Berkeley there are two degreed programs, one strongly influenced by the engineering faculty and the other strongly influenced by a sense of liberal education, Patt said. He believes that this arrangement is appropriate for Berkeley. "If an employer wants to know that he is hiring a graduate of an accredited program, then he should know that the graduate has passed a minimal standard of both quality and scope."

"It is no disgrace," he concluded, "if a department offers two separate degreed programs. If you want to have exempt status at Boeing, you take this program. If you want to learn something with intellectual content, you take this other curriculum. And if you want to, you can stay in school for additional years and take them both."

Truth in advertising. "I hope that Gene (Lawler) and Yale (Patt) are advising their students so that they can make a conscious choice when they enter one program or the other of what its advantages and limitations are," Booth countered. "The student who goes through the BS program of engineering education has a significantly different preparation, education, and goals from the student who chooses the BA program.

"I hope Gene is telling the BA student that if he wants to compete with the BS student that he might have to take another year of professional education, possibly at the master's level, to come up to the same level of professionalism," Booth continued. "Of course, the BA student, in addition, will have a broader education that will allow him or her to become a more rounded individual. What we are talking about in accreditation is truth in advertising."

Further comments. Following the initial presentations by the panel members, moderator Cain encouraged more than an hour of comments from the floor and further elaboration of their views by the panelists.

"Schools invite you in," said one knowledgeable member of the audience of the accreditation process. "I've never had anybody complain that there is a mountain of paperwork. It's never been a negative experience for me. We don't go in to offer advice. It's not our job to tell you how to run your programs. It's like a litmus test: you've got it or you haven't.

"A lot of people take the attitude that they don't need accreditation," he continued. "It may be that CSAB will die a natural death, but if the experiment is never run, no one will ever know. It will help the little schools who are being pressured by their administrators to offer computer science programs because there is money to be made. People in these schools need something like CSAB so they can say to their infrastructures that you can't expect us to run a computer science program without sufficient resources."

The top 100 schools may not need an accreditation process, but what are things like at the 500 or 600 level or even at the 1199 level (out of the 1200 schools offering some kind of computer science), another member of the audience asked.

"Some of these schools have never gotten off their campuses to compare what they are doing to the norm," Booth replied. "In some cases they are completely unaware of how inadequate their programs are."

In such cases the "mountain of paperwork" actually becomes an instructive self-evaluation tool that challenges the faculty and administration to take off their rose-colored glasses and see how they match up with the criteria. They start facing up to some of the embarrassing questions they may have been avoiding. In several cases they have put off the accreditation visit phase of the process.

"I would also like to point out," Booth continued, "that some of the self-appointed top 100 schools may have excellent graduate programs but are neglecting their undergraduate programs. Students who go to these schools because of their reputations get seriously shortchanged."

Sometimes these schools are quite embarrassed when an accreditation team looks into their undergraduate program.

Society needs well rounded individuals—the goal of the liberal arts—to help carry on civilization as a whole. It also needs professionally qualified individuals—the goal of computer science accreditation—to perform some of the tasks on which civilization rests. Moreover, young people have different financial resources and time-to-job horizons. And so the debate continues. ☐

Ware Myers has been a Contributing Editor for the Society's magazines since 1976.

中國科學教育

Special Feature

Computer Science Education in the People's Republic of China

John H. Maier

It is to the benefit of the United States to welcome China both as a trading partner and as a new young member of the computer-based electronic village community.

The United States now does more trade with Asia than with Europe, a trend that is predicted to continue into the next century. The United States is now China's largest import partner, and third largest export partner, after Japan and Hong Kong. As a nonaligned, friendly, developing country, China wants to modernize, and, in fact, catch up with Western developed countries. A key to modernization for China is computer technology, and, as a corollary, computer science education, but success will not occur in one great leap. In addition to China's domestic efforts, China now looks to the West in its efforts to join the information technology community. During the years 1983-85 China has bought over a half billion of US-manufactured computer hardware. It is to the benefit of the United States to welcome China as both a trading partner and as a new young member of the computer-based electronic village community. This article explores China's recent educational activities in computer science, and is current through the fall of 1985.

Prior to the 1960's computer-related education in the United States emphasized mathematics and engineering and was mainly limited to those who had design and on-hands experience with computers. By the mid-1960's, spurred partly by the Sputnik push for advancements in science education, partly by the realization of the growing interdisciplinary influences of computer technology, and partly by the maturation of computer science as a discipline, specific computer science, or CS, academic curricula began to formalize in American universities. At about this time, two important reports*—the National Academy of Sciences' "Digital Computer Needs in Universities and Colleges (1966)" and the President's Science Advisory Committee's "Computers in Higher Education (1967)"—resulted in specific national curriculum recommendations called Curriculum 68 (later updated to Curriculum 78). As early as 1962 Stanford University had already established a CS department, one of the first, and by the latter 1960's CS departments had begun to open up on campuses across the United States.

As reported in a survey conducted in 1977, approximately 3000 US educational institutions offered over 1500 computer-

---

*Further information can be obtained from the National Academy of Sciences, 2101 Constitution Ave., Washington, DC 20418.

0018-9162/86/0600-0050$01.00 © 1986 IEEE

related degree-granting programs, and of the 3000 over 100 offered PhD degrees in computer science. Also, 130,000 students were specializing in computer science (80 percent undergraduates).

In 1982 30 CS department administrators met to review the CS education needs of the United States. They predicted, as did a then-current White House Office of Science and Technology Policy report, that despite substantial growth in American CS education—250 new PhD's in 1979—there would still be an acute CS professional shortage into the 1990's in the United States. They noted that explosive industrial and information technology growth was not being matched by the output of US educational institutions.

China's computer education crisis

The total number of computer professionals in China (in 1984 estimated at about 100,000) is still fewer than can be found within a healthy radius of Stanford University, in Palo Alto, California. China's top leaders, including the Minister for Education, the President of the Chinese Academy of Sciences, and leading members of the State Council, all realize China's urgent need to join the information technology revolution. Members of the Chinese Academy of Sciences have begun to share the developed nations' vision of the world as a global village with an electronic system of interconnection.[1] President Reagan, in his 1984 visit to China, lectured at Fudan University in Shanghai and mentioned a new US-made computer being installed there. In fact, between 15 and 30 Chinese universities are receiving Honeywell upper-medium-scale machines. Despite this general awareness of China's pressing need for computer-related technology, and computer education, a close look at what is occurring there today reveals a discouraging situation, one in which China's attempt to meet modernization goals and to "catch up" with countries who possess a technology that is going through generational changes every five years is not moving fast enough.[2]

China, a country of more than a billion people (75 percent of whom are involved in agriculture), is presently decentralizing large portions of its centrally planned economy and giving rein to market forces. China's needs are unique to China, but there can be no doubt that computer technology and related communications technologies are central to China's hopes for modernization.

This article is a result of a year of lecturing in China in a special program of introducing computer science to recently graduated Chinese scientists from almost all parts of China, and from many of the key universities in China (these students were being prepared for graduate doctoral studies in the United States), and of visiting computer centers and CS science departments in 16 key universities in China.

A department store in busy Guangzhou.

A brief overview of China's educational system

There are presently over 800 universities and colleges, 100,000 secondary schools, and 860,000 primary schools in China, with 1 million, 40 million, and 136 million students, respectively. Of the 800 universities and colleges 97 have been designated as key universities by the Ministry of Education, and among them 10 or 15 stand at the top as the nation's best higher education institutions. Chinese students pursuing higher education comprise 0.001 percent of the total population and 0.007 percent of its total student population. In Chinese universities tuition, lodging, and medical care are free. Students pay for only their food and textbooks. Entrance to most universities is by examination. In 1981 institutions of higher learning began to grant bachelor, master, and doctorate degrees after 10 years of the educational chaos resulting from the Cultural Revolution. Since 1981 China has produced 18 doctorate, 15,000 master degree, and 320,000 bachelor degree holders.

Lecturing experience in China

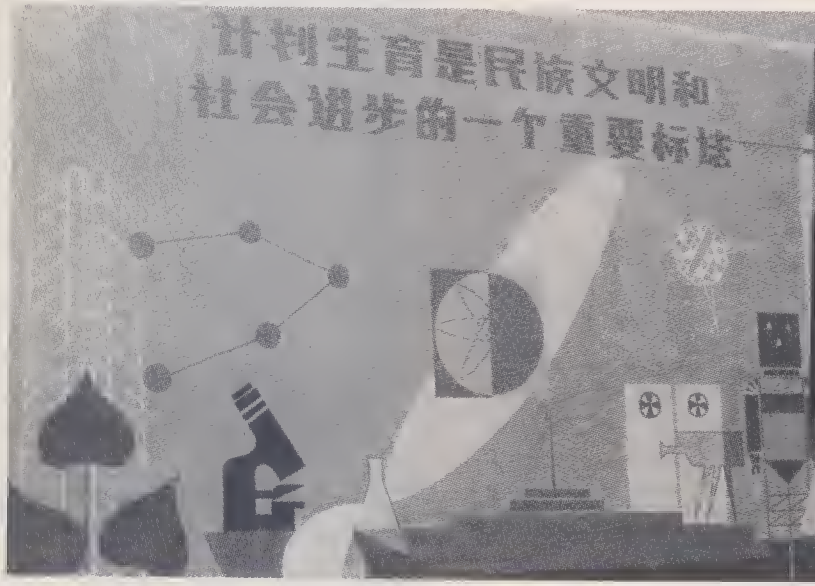
A special adjunct faculty professorship was established at Sun Yat-sen University in Guangzhou during the 1984-85 academic year in the belief that computer science was the discipline that would be a prerequisite for successful post-graduate studies in the United States.

Students intended for such studies were divided into three groups according to how much CS experience and education they had. Testing their ability to define selected CS terms and inventorying their hardware and software experience resulted in the following breakdown:

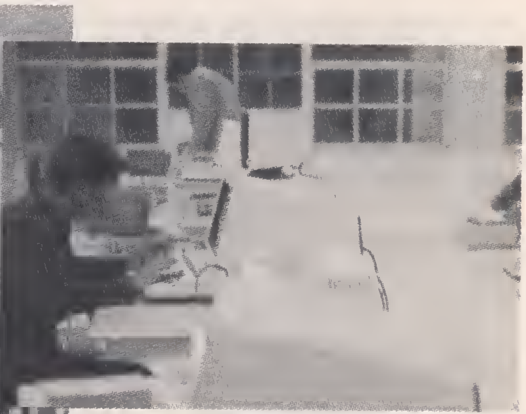
- Group I (approximately 70 percent) —little or no prior experience.
- Group II (25 percent)—some, but limited, prior programming experience.
- Group III (5 percent)—prior advanced scientific computer-related experience, generally in applications software.

The seminar given to Group I students was titled "A General Introduction to Computer Science"; to the Group II students, "A Survey of High-level Procedure-oriented Languages"; and to the Group III students, "Advanced Computer Applications." Some of the topics in the latter seminar were:

- Computer Performance Measurement,
- Computer Technology—A State-of-the-Art Overview,
- Procedure-oriented Languages,
- Computer Science Education in the United States,



Billboards are an important means of mass communication in China.



A new computer installation at Nanjing University.

- Fifth-generation Goals,
- Voice Recognition Systems,
- Space Orbit Control Software,
- Natural Language Processing,
- Expert Systems, and others

In general, the young, recent university graduates taking these seminars were enthusiastic, receptive to using computers within their disciplines, eager for hands-on experience, and open to promoting computer science. A central feature of all course material was to prepare the students for the massively pervasive presence of computer technology in the United States. It is worth noting that some of this course content in the format presented (a general survey of computer science focusing on information processing, not number crunching) was available nowhere else on campus, largely because the previous generation of educators in China does not have the background.

Campus automation in China

At best, campus automation in China is only beginning, and university officials are going through the same painful processes that US campuses experienced in the 1960's, when campus administrators were just beginning to decide what role this technology should play on campus.

Recently, on a selected number of key Chinese university campuses, computer centers independent of the CS department, and reporting directly to the university president, have been established—a setup similar to that found on American campuses. This occurred when it was realized that computer technology was to be, in fact, a campus-wide resource. Management of this new resource, however, requires that the campus be advised of the computer's potential to improve both teaching and research. As presently configured on campuses in China, where new fairly large machines have recently been installed, with for example perhaps 60 to 70 remote terminals, all of the terminal workstations have been placed very close to the computer center building rather than in dispersed locations—again, a configuration found on many US campuses in the late sixties and early seventies. Generally, such an approach tends to inhibit the rapid infusion of the technology into the infrastructure of the various departmental faculties and student populations.

China has a long way to go to match the current environment of US campuses, on which campus-wide local area networks deliver computer services directly to user populations within their work environments, but now is the time for systems planners to do so. Chinese campuses in selected key universities are undertaking substantial building expansion programs—certainly a healthy sign. But constructing new buildings meets the form not the content of future technological needs, an important one being campus-wide delivery of computer services.

Automation of Chinese libraries

A critical campus need and application is library automation, a seminal information technology accomplishment. Libraries are the heart, soul, and key resource of any learning and knowledge production community. This has been true since the time of Socrates* and the Hanlin Academy of the Han Dynasty, but in the twentieth century the worldwide explosive growth in information mandates computerization of important libraries, something that occurred in the United States in the mid-1970's. An important benefit, it should be noted, of automation is that it necessitates rationalization and rigorous

---

* "Socrates," interestingly, is the name of Stanford University's campus-wide library automation and information retrieval system. It has always been this author's desire to sit at the desk and be able to access any library in the world, regardless of language. It should be noted that in China national automated research centers and libraries are high priority, but accomplishments have been slow.

Provincial museum in the suburbs of Nanjing, an urban center west of Shanghai.

systemization, a healthy process that would be instrumental in reforming and modernizing China's library systems.

Difficulties with automating Chinese characters have essentially been solved using many viable systems (in varying hardware and software environments). In fact, Chinese characters (ideograms) input and output methodologies can differ according to circumstances, needs, and levels of skill of user populations, and the main focus should now be on internal standardized systemic binary representation so that there is compatibility with many system interfaces, including those of Western products. The technology is available; choice is now the appropriate action.

China's library classification coding system is dated (for example, it lacks categories for many newer technologies in computer science), and differs from other widely used systems such as the Library of Congress system and the Dewey Decimal System, but it is possible to work around this situation with software system design that is flexible enough to allow the classification system to grow and expand, and to allow tables to be cross-referenced into other coding classifications. At Beijing Library, which is the national library of China, an automation project on a Fujitsu is making progress, and cooperation with the US Library of Congress has been established so that Beijing Library will be using MARC format computer records for interlibrary exchanges. China's 1978 National Science and Technology Conference, attended by Deng Xiaoping, gave top priority nationally to library automation. Progress has been very slow.

The knowledge base contained in China's libraries is, in part, integral to general world scientific knowledge, but it also contains highly specialized categories of knowledge peculiar to China. However, as of the fall of 1985 only one small, limited, prototype library automation project in China has come up.

The point is that the technology now makes it feasible for China to automate its important libraries, but until this watershed is reached, China's researchers, scientists, students, and faculties will be seriously handicapped. Further, until China's academic communities become familiar and comfortable with on-line information technology, it will not penetrate other sectors of China's economy and society.

Campus management information systems

Another area of campus-related automation is Management Information Systems, or MIS. I have encountered only one Chinese campus on which development of such a system was underway. Systems now taken for granted on US campuses (for student registration, course listing, grades, alumni listing, transcripts, budgeting, etc.) no Chinese campus has yet implemented. The rigid structure of, and lack of cooperation between, working, or organizational, units on Chinese campuses, however, could hinder such a venture. A lesson that has to be learned is that to be effective computerization must cut across all bureaucratic lines. Implementation of an effective campus MIS would result in a first for China—one that could be replicated on an interuniversity consultancy basis, thus saving China important applications developmental time redundantly expended elsewhere. And finally, automation will not occur easily elsewhere in Chinese society until campuses get their own houses in order.

CS education

China's activities in computer technology date from 1958, when the Institute of Computing Technology under the Chinese Academy of Sciences produced a first-generation vacuum-tube computer capable of 1000 operations per second. Today, China has successfully built a 100 million operations per second vector array processor, the "Galaxy," or "Yinhe," placing China among the few countries in the world that can produce supercomputers. China has also successfully placed into geostationary orbit a functioning communications satellite. But China has not yet matched these salient accomplishments, which required intensive research and development coordination, with CS activity of breadth and depth in China's various social and economic infrastructures.

It could be that, as of now, the Chinese are beginning to produce a new generation of broadly trained CS professionals. For example, thousands of Chinese students, not all in computer science, are studying in the United States, and when they return to China, they will be more familiar, to one degree or another, with computer systems after their encounters with the all-pervasive computer in the States. The number of these professionals now being trained is, however, inadequate for China's future needs.

Vis-a-vis computer science in China, the universities most frequently mentioned are: Qinghua, Jilin, Beijing, Nanjing, Fudan, Shanghai Jiaotong, and Xian Jiaotong. However, CS departments are rapidly emerging in many other universities, including, for example, Hunan University, with its close ties to China National Defense University, where the Galaxy was developed. Hunan University, for example offers such courses as Discrete Math, Digital Logic, Fundamental Project Design, Data Structures, Compiler Principles, Operating Systems, Microcomputer Applications, DBMS Structures, Methods of Design and Analysis, Image Processing, Special Engineering, LISP, Technical Interface LSI, Pascal, C, Fortran, Networks, Computability, CAD, AI, Expert Systems, Software Engineering, Fast Fourier Transforms, Unix Operating System, Control Systems, Chinese Information Systems, and others.

Some schools have extended strength in such applications areas as physics, mathematics, astronomy, space sciences, engineering, and other sciences. In general there are about 15 universities that are invariably considered to be tops in computer science in China. In each school I visited, I held discussions with the president, the department administrator, faculty, and, last but not least, students. From these talks I gleaned that many more schools are planning to upgrade their computer resources by acquiring hardware, expanding CS curricula, training faculty to teach computer science or integrate it into other disciplines, and increasing student recruitment. However, it would seem that, as a first priority, China should concentrate on strengthening and reforming the current CS departments where substantive and productive work is occurring.

Because a thorough treatment would require an entire other article, I cannot give an in-depth description of the Chinese Academy of Sciences computer-related institutes, or of the computer-related activities of governmental ministries, provincial governments, etc.

A picture of China's top-level CS departments has not fully emerged, but it is possible to discern both encouraging developments and problem areas.

Some encouraging developments are:

- The emergence of AI studies,*
- A healthy interest in the computer-related activities of colleagues in other countries, especially in the United States,
- The high level of student enthusiasm for computer studies (which should not be underemphasized and which might indicate in Kuhnian[3] terms that the previous generation is on the twilight side of a new paradigm for China now unfolding),
- The relatively rapid growth of undergraduate enrollments in CS studies (top schools now have CS department enrollments ranging from 200 to 500 students),
- The creation of graduate CS programs (though only one or two departments in China confer the PhD in computer science),
- The excellence of some existing programs (though they do not match the strength of those offered by top US schools),
- The growth and quality of Chinese journals in the field,
- The creation of new applications and substantive student projects,
- The emergence of user groups, and
- A broader awareness among students and a growing awareness in the general population of microprocessor technology.

Also, some CS libraries are fairly well supplied with Western and other non-Asian journals and books, although the journals reach these outlets six months or so after publication. Nonetheless the list of titles is impressive; unfortunately, students are not given access to them.

Some problem areas are:

- The *danwei* cultural attitude that results in noncooperation among various working units,
- The "inbreeding" system in universities in which the graduates of school become faculty of the same school and few outsiders are hired,
- Excessive concentration on theoretical work with a slow growth in applications work,
- Insufficient attention to and effort in interdisciplinary research and applications,

Sampling of CS courses being given in some Chinese universities

Beijing University. Research in laser photocomposition of Chinese character texts, image processing, data acquisition, DBMS development.

Nanjing University. Programming language design (XCY-2), distributed system software (distributed C language and distributed dBase II), Chinese word processing software, library automation, development of an Ada compiler.

Huazhong University. Campus automation.

Xian Jiaotong University. Development of an Apple II Lisp compiler.

People's University of China. Development of a relational DBMS on an HP 3000, Chinese information processing systems, development of a system for national economic and statistical data.

Wuhan University. Library automation, formation of an interdisciplinary AI group.

Qinghua University. Development of applications software packages for microcomputer and minicomputers, software product testing tools.

Hunan University. Final product implementation of PDP 11/23 Chinese word processing and text editing, conversion of system to IBM-PC (in Pascal), communication of system with IBM 43XX.

Shanghai Jiaotong University. Software for financial analysis, graphics for microcomputer and minicomputers.

Jilin University. Research and development of expert systems, AI work.

Also of interest at the Beijing Institute of Computing Technology of the Chinese Academy of Sciences, work was in progress in development of Lisp, C, and Prolog compilers on the Chinese BCM-3 machine.

---

* The Chinese are researching in natural language processing, voice systems, knowledge representation, visual systems, robotics, etc. In China Jilin University is always mentioned first for AI work. At this school Professor Wang Xianghao is noted and empowered to confer the PhD. In November 1984 China sent a delegation to Japan's Fifth-Generation Conference. The AI Journal *AI World* is published by Xinhua Communications, and Chinese researchers are publishing abroad in, for example, *The AI Magazine.* The Chinese have a traditional medicine expert system.

- Few Chinese texts on the subject (In visiting many bookstores I found at best not more than 100 to 200 titles in Chinese on computers, and many were recent translations from texts in English.),
- A paucity of information on courses offered at universities (Universities do not publish campus-wide curriculum catalogs, and departments tend to hold on tightly to their course listings, making it difficult, for example, for a biology student to take a CS course.),
- Underestimation of the role of software engineering and economics, and
- Lack of a clear vision of where the discipline of computer science should lead China in the future.

Please note that the above are only impressions—though strong ones.

If I had a mandate to reorganize and improve CS education in China, my recommendations would include

- Develop national CS curriculum guidelines, such as occurred in the United States in the 1960's;
- Mandate that faculties periodically accept teaching and research positions on a regular basis at other than home institutions;
- Give higher priority in the national budget to CS departments including monetary incentives for faculty;
- Design elective courses in computer science for students majoring in other fields;
- Openly publish department curricular offerings;
- Open up CS department courses to nondepartmental students on a CS department consent basis;
- Involve students in real projects such as the development of campus management information systems;
- Ask CS departments to submit five-year plans that are both quantitative and qualitative and that involve a vision of the campus of the future and China's need; and
- Establish visiting professorships for faculty from abroad.

Additionally, in a technology in which there is an acute shortage of a work force faculty members, and even students, in China should be encouraged to serve as consultants to organizations outside of the university. In China this would be good "theory and practice."

Into the future China will suffer from a shortage of computer professionals, a situation now occurring in the United States—only China's shortfall will be greater if present levels of CS education continue. According to statistics announced by China, China has approximately 100,000 computer professionals, and information is not available on their levels of skill. Even if the top 15 CS departments enroll 400 students each for the next decade on a continuous level, and 100 other schools maintain a CS enrollment level of 100 students each, this combined effort will result in only 35,000 additional CS professionals by 1995. At this rate China will be graduating one CS student for every 250,000 people in China. Currently, the United States graduates one for every 4000 people.

In just one instance of how it is lagging in technology, China has not yet implemented its first national digital network, the difficulty being the quality of terrestrial phone lines in China. In the United States people take for granted a plethora of networks, public and private, and do not hesitate when they hook up a modem to the nearest phone. China, however, is now importing such technology, and is negotiating now with the United States for an advanced geostationary communications satellite. This may solve China's national needs for reliable data communications, and Chinese computer scientists may then get to work on a long list of applications waiting to be implemented. But everything takes time.

China wants to modernize by the year 2000, the year in which it would like to host the Olympics, to a level of the advanced countries in the 1980's. It must implement plans not only to catch up with technology, but also to stay abreast and move ahead of technology. China is now in the "catch up" phase in a very dynamic technology. China must move as fast as the technology just to maintain its current relative position; to begin to catch up, or make relative progress, China must move faster than the technology itself. Computer science is a key technology in the sense that it is an integral part of all the sciences today; a full realization of its importance will become evident only in the future.

> China must move as fast as the technology just to maintain its current relative position.

Perhaps a national computer conference addressing China's CS needs (similar to China's 1978 National Science and Technology Conference) involving computer scientists and CS department administrators, university presidents, Ministry of Education officials, and Chinese Academy of Sciences members, and with the blessing of the State Council, would suffice and to formulate clearly what China must do to bring its CS technology up to par with the developed countries.

If there is doubt as to the importance of computer science to China's modernization aspiration, then I will quote the Tang Dynasty poet Li Bai:

The woods are black and a wind
assails the grasses
Yet the general tries night archery—
And the next morning he finds his white
plumed arrow
Pointed deep in the hard rock

This poem—a Zen-like koan—answers the question "How is modernity achieved in China?" thus: Though shot in the black of night, the arrow went straight to the target, much as the seemingly aimless and general process of computerizing China goes straight to the heart and essence of modernizing China.

As a final note, US technology export guidelines for China are still atavistic, are seriously hampering China's modernization efforts, and are not appropriate to the nonaligned, friendly relationship between China and the United States. The United States, with its massive inventory of hardware and software, and well-developed systems, should not feel any threat from China's relatively modest efforts to develop computer-based systems for its critical energy, transportation, banking, tourism, management, education, and manufacturing infrastructures.

Implementation nationally, provincially, and locally of computer-based information technology in China has the highest and strongest support of China's leadership. To quote Professor Song Jian, Chairman of China's State Science and Technology Commission:[4]

> I would like to imagine that some decades in the future China will be inextricably involved and locked into the complicated international economic, science and technology systems of the world. At that point no one would dare draw back from the world systems into seclusion again.

References

1. Xin Jishu Geming De Jueqi, *The Abrupt Rise of the New Technology Revolution,* Chinese Academy of Sciences Publications, 1984; also, the Chinese translation of John Naisbitt, *Megatrends: Ten New Directions Transforming Our Lives,* China Social Science Publications, Beijing, 1984.
2. Shi Jie Xin Jishu Geming, "Modernization Information Collection," *A Dictionary of the World's New Technology Revolution,* Popular Science Publishers, Beijing, 1984.
3. Thomas S. Kuhn, *The Structure of Scientific Revolution,* University of Chicago Press, 1972.
4. *China's Changing Environment for Science: Report of a Trip by Members of the AAAS Board of Directors* (*31 August-16 September 1985*), American Association for the Advancement of Science, p. 4.

John H. Maier is currently a senior independent computer science consultant. Invited to the People's Republic of China by the Ministry of Education, Maier lectured as an adjunct professor of computer science for an academic year (1984-85) at Sun Yat-sen University in Guangzhou, and traveled in eleven provinces visiting institutes, universities, and computer centers. He was previously with the Bank of America, and the Agency for International Development, and the US Air Force.

Maier received the BA from the University of Hawaii and the MA from the East-West Center/University of Hawaii.

Readers may write to Maier at 8830 Sandcastle Court, Fort Worth, TX 76179.

New from Springer-Verlag . . .

Reliable Sources of Computer Science Information

New

Reliable Computer Systems

Edited by **S.K. Shrivastava**

The purpose of this book is to classify and collect the papers from a research project at the University of Newcastle-upon-Tyne which investigates the design of highly reliable computer systems. Included are discussions of system reliability, principles of fault tolerance, recovery in concurrent and multilevel systems, and distributed systems.

1985/580 pp/215 illus/Cloth $39.50
Texts and Monographs in Computer Science
ISBN: 0-387-15256-3

To order, please write to:
Springer-Verlag New York, Inc.,
Attn: G. Kiely,
175 Fifth Ave.,
New York, NY 10010
or visit your local academic bookstore.

New

On Knowledge Base Management Systems

Integrating Artificial Intelligence and Database Technologies

Edited by **M.L. Brodie** and **J. Mylopoulos**

Contents: Knowledge Base Management Systems. Knowledge Bases versus Databases. Retrieval/Interface/Reasoning. Extending Database Management Systems. Extending Knowledge-Based Systems. Knowledge-Based System Design Issues. Advanced Hardware for Knowledge-Based Systems. Epilogue. References. Index.

1986/Cloth/Price not yet set
Topics in Information Systems

Springer-Verlag
New York Berlin Heidelberg Tokyo

Also available in the series Topics in Information Systems . . .

Office Automation

Concepts and Tools

Edited by **D. Tsichritzis**

1985/444 pp/86 illus/Cloth $29.50
ISBN: 0-387-15129-X

Query Processing in Database Systems

Edited by **W. Kim**, **D.S. Reiner** and **D.S. Batory**

1985/370 pp/127 illus/Cloth $32.50
ISBN: 0-387-13831-5

On Conceptual Modelling

Perspectives from Artificial Intelligence, Databases and Programming Languages

Edited by **M.L. Brodie**, **J. Mylopoulos** and **J.W. Schmidt**

1985/510 pp/25 illus/Cloth $34.00
ISBN: 0-387-90842-0

Reader Service Number 3

ソフトウェアー

Special Feature

The Japanese and software: Is it a good match?

Laszlo A. (Les) Belady
Microelectronics and Computer Technology Corporation*

The key to the Japanese approach to software development is the separation of the planning and implementation phases.

This article is a purely personal account, based on 18 months in Tokyo in a research organization consisting entirely of Japanese-speaking personnel, except for myself. It is not intended to be a systematic or scholarly study of the Japanese software industry.

I had the good fortune to have an assignment within IBM to go to Japan to help Hisashi Kobayashi, director of the IBM Japan Science Institute, form a software technology research group. This institute is the newest member, now four years old, of IBM's worldwide research organization.

My initial intention was to learn the Japanese language so as to be able to participate fully in the work. I already knew four languages fluently, including my native Hungarian. Since Hungarian does not have the same linguistic roots as the western European languages, I thought I would have a head start in learning Japanese. Alas, it didn't work out. I did learn about 400 Chinese characters and I did become able to converse with my Japanese language teacher after a fashion, but I did not become fluent enough to talk with my coworkers in Japanese. That was a disappointment.

First, let me give you a brief rundown of the Japanese software industry. Perhaps the best known activity is the Fifth Generation Computer Project. It is primarily an architectural effort, but it includes the development of the software necessary to make the hardware complete. I don't believe this project is primarily aimed at improving the process of efficiently producing high-performance software. That is another issue. Still, ICOT, the institute managing the project, tends to be the gatekeeper between Japan and the rest of the world on the technologies having to do with computers.

The real down-to-earth software technology effort resides in the Big Three or the Big Six companies. This effort is aimed at improving productivity and quality in an industrial every-day sense. The Big Three are the Nippon Electric Co., Hitachi, and Fujitsu, and the other three are probably Toshiba, Oki, and Mitsubishi.

In addition, there is a large effort, called the Sigma project, involving a combination of major customers and software houses. The effort started with a couple of dozen participants and is now close to 100. It is being standardized around Unix as the basis for developing software and is tying the participants together through a network. The participants will be able to exchange tools and techniques over this network.

How about the universities? In the United States Americans pride themselves on the excellent relations between industry and academia. In Japan it is my intuition

---

*Condensed by Ware Myers, Contributing Editor, from the plenary address at Compcon Spring, March 6, 1986. The Japanese Kana translation of "Software" was provided by CompuBot, Los Angeles.

0018-9162/86/0600-0057$01.00 © 1986 IEEE

Les Belady with Japanese computer scientists in Tokyo.

that the universities are not much tied to industry except in the case of the Fifth Generation architecture. In software technology I regret to say that I did not see much cooperation or easy transfer of technology from universities to industry, or vice versa.

In general, however, I found a very positive attitude toward software. The strength of this attitude can be illustrated by an experience I had several years ago. I had been invited to spend a day at the Toshiba software factory by a friend, a researcher there. I felt that I was a rather insignificant software researcher, yet I was received by the plant manager and his entourage—and this was a large plant with 20,000 employees. It manufactured process-control equipment for steel mills, power plants, and other industries.

All of them spoke quite good English and I'll never forget what the plant manager said: "We are old-time engineers, building power plant and control equipment. Personally I don't know anything about software. I know only one thing: it is important. It is important for the future. It is important for Japan. Therefore, we do it and we support it."

I have yet to hear top executives in the United States speak so positively. I hear them saying something like, "I wish software would disappear, because it is just a problem for us."

In Japan, in contrast, I found this positive attitude everywhere. Top people do not necessarily understand software in all its details, but they do understand that it must be done properly.

Japanese education

The "sensei" culture places great emphasis on the teacher. The student-teacher bond is a relationship for life. For example, my teacher of Japanese, who is 69, cancelled our class one day because he had to meet with his teacher, who was in his high eighties. Similarly, my teacher is my friend for life. The Japanese have a love of teaching, as well as learning.

The consequence is that education is taken seriously. Mathematical courses in high school are a must, for example. The Japanese are good engineers. Their per capita production of engineers is twice that of the United States. Incidentally, Americans can pride themselves on producing far more lawyers per capita!

One Japanese practice that aggravated me at the time was the subjection of university graduates to as much as a year of additional full-time classroom instruction when they joined a company. At IBM Japan we spent six months, ignoring the fact that these young people had just come out of the classroom.

After a while I think I got the idea. What they learn in college are skills. What they don't learn there are real-life problems, the problems of the particular company. The company trains them in these specifics and only then are they expected to become productive.

The new graduates recently seem to be becoming a little rebellious. In college they have been learning new and exciting things—perhaps executable specifications or formal verification of correctness. When they come to a research organization like the IBM Science Center, they expect to continue this "far out" work. Instead they are taught subjects like real operating systems and other practical matters. They find that the actual software world and its problems are completely different from what they expected.

Nevertheless, in Japan industry provides additional education. They do not accept the idea that young people just out of school should immediately start working and producing.

Separation of planning and implementation

In Japan if you want to get your people to start right to work on a problem, you can't. At first I found that aggravating, too. If I insisted that they attack a problem, people would actually slow me down. "Wait a minute," they would say, "we have to learn first." They would get all the relevant books and papers. They might spend weeks, perhaps months, learning about the problem.

Planning and implementation are sharply distinguished activities in Japan. This observation is mine. I never heard it stated in this fashion, but I think it is true. In the United States, of course, there are great improvisors—one never knows whether they are planning or actually building something. In Japan the two modes are distinct. Either you are planning or you are implementing.

Planning. The planning mode is quite informal. In particular, there is no rush. This seems to be where many of the reports by Westerners come from, that Japanese meetings are slow and tedious, not fast and action-oriented as in the West. The implementation mode, however, can be faster and more efficient. Then there is no need to deliberate because the plan has already been set.

In the United States the work of planning and the work of implementation are specialized. Each type of work is performed by different people and it is rare to see much interaction between them. In Japan, however, the participants in the planning phase are quite often the same ones who will execute the plan.

Being in no rush during the planning phase, the Japanese have time to gather lots of information. They have time to explore lots of alternatives. They have time to spend on what Americans might regard as details. Procedures come very slowly because they examine carefully a number of courses of action before they settle on one.

It is not true that everybody in Japan believes the same things. That is an impossibility. What is true is that they listen to each other. They have time to reach an understanding of other people's points of view. They have time to understand the context in which the plan is to operate and to understand the plan itself thoroughly. Detail understanding is the goal, not consensus for its own sake. The planning becomes an educational process in itself.

I have the feeling—although I have no solid evidence for it—that the Japanese have a built-in desire to plan things. In contrast, you might say that Americans have a built-in desire to act.

Implementation. At the end of the planning phase comes the time when tasks are assigned. The tasks are not necessarily broken out by occupational specialties; they are more likely to be sorted out in terms of problems.

The Japanese can work this way because they are oriented to getting the job done. Even people who appear to be overqualified for a little problem are willing to work on what Americans might regard as an assignment beneath them. In fact, one of my Japanese friends provided an example of this attitude.

"Here is another report we have to put together," I said to him apologetically. "You remember that you did it before. I am afraid that you are going to have to do it again."

"Wait a minute," he replied. "What do you mean? Why are you apologizing?"

"It is such a mindless thing for a person of your capabilities to have to do," I said. "It will bore you."

"I don't mind. It doesn't matter," he said. "It's a job to be done. It doesn't have to be intellectually challenging."

In making job assignments, however, the Japanese demand that you be as detailed as possible. In my casual Western style of management I often did not do that. I left things to the imagination; I thought things were implied. In Japan you have to be specific, because they take you literally.

I learned this fact of life very rapidly from my secretary. I gave her a job one afternoon which I knew would spread over several days. When I returned after dinner, she was still working on it—she ordinarily left at six o'clock. It seems that I had neglected to tell her that it was not urgent. She had taken my instructions literally—do it now. I should have said, work on it this afternoon, go home at six o'clock, continue tomorrow during regular working hours until you have finished.

Implementation in this way may be to the American way of thinking a little bit on the mindless side. It sounds almost like programming a computer in assembly language. To some extent it is mindless, except for one thing. While efficiency in implementation is a major goal, they will not do the work sloppily just to keep to deadlines. They will take the time to do it right.

The important point is the work is well planned. Everybody knows what to do. They know what kind of software they are doing.

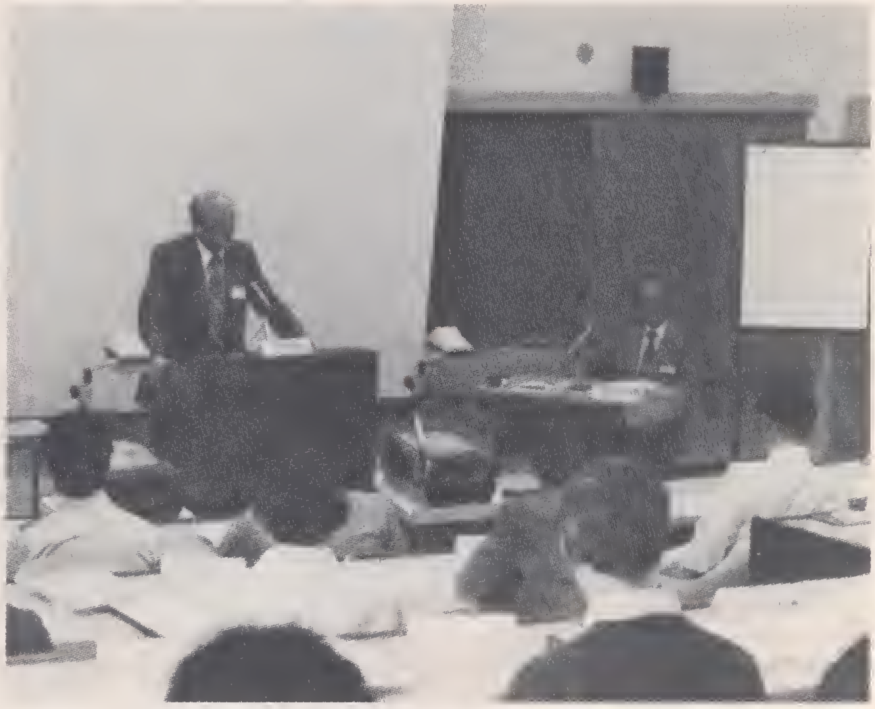
> Procedures come very slowly because the Japanese examine carefully a number of courses of action before they settle on one.

In fact, the Japanese software factory goes beyond that by providing human feedback. It monitors errors, collects error rate data, and distributes the errors over different modules. Then it goes beyond just presenting the resulting charts to management. It feeds the error data back into the process itself. Errors in such and such a module done by such and such a team are fed back to the team. It is feedback in a human sense.

When the software process is well planned and designed, it can accommodate people with much less preparation and schooling than it needs when implementation has to be interspersed with innovation because the planning was not complete. Someone who just knows how to do a few things well in a coding sense cannot be expected to make successful improvisations to take the place of poor planning.

Japanese attitudes

That brings us to the issue of whether the Japanese are creative or not. Americans have said they are noncreative so loudly and so often that they have taken it as a challenge. Two years ago, while I was there, a movement developed to help people come up with fresh ideas, to induce them to invent. That itself became a planned activity. One of the results of this movement is the Basic Technology Research Promotion Center, where people dream, do far out things, and come up with unusual solutions. You would agree with me, if you could see some of the Japanese gadgets, that creativity is



The author giving a lecture at an IBM conference in Japan.

palpable in many aspects of the country's life, not only artistically, but also industrially.

In fact, the Japanese particularly love new gadgets, even though in many ways it is still a very traditional country. I myself am usually quite satisfied with an old car – I still have one from 1965. Many Americans are like me. The Japanese usually change cars more often than that. There is a fascination with mechanical toys and robots and practical gadgetry. This attitude extends to automation. This underlying love of the new is a likely contributor to creativity.

Problem orientation. The Japanese problem-oriented attitude impressed me again and again. They are problem-driven. They consider themselves to be employed to solve the problems of their organization, not just to show off their muscles by doing well in their particular area of expertise. Doing well in a specialty is irrelevant. The relevant consideration is how to apply your abilities to solve the problem you now face.

In the United States I might ask an employee to tackle an administrative problem. He says, "No, I am a researcher."

I might ask another employee to work on this obvious organizational problem, reminding him that, "You yourself called my attention to it." He says, "No, I am an administrative assistant. I don't do that."

In Japan solving these problems would have been much easier. We would have had a meeting and discussed the two problems. Very soon we would have agreed on what the problems were and I would have said, "You do this one, and you over there do that one."

That would have been the end of my management problem—no further energy would have to be expended. Once something is perceived as a problem, it becomes clear that it has to be solved. Then it is immaterial who is doing the solving. That is quite different from the American way.

Motivation. The high Japanese motivation may arise from the circumstance that Japan is a country poorly endowed in natural resources. In the absence of imports, it would run out of energy in a few days. It would run out of food in a few weeks. It must manufacture and trade in order to live. Its chief asset is its people and their motivation.

Willingness to cooperate. To the Japanese the contribution he can make to his organization is the main source of his satisfaction. He takes pride in the progress of his group. He is more likely to spread his know-how selflessly among his co-workers than an American is.

Because of the Kanji character problem, very little written communication is possible at the quick notification level, even with computers available. Handwriting is tedious and slow. Hence, there is lots of oral communication in a Japanese organization. This intense oral interaction is, I believe, conducive to cooperation.

The Japanese usually work in open offices without dividing lines. This arrangement is not a matter of poor construction; rather, it is there to enable people to constantly ask questions of each other. Also they share files with each other, as well as a passion for neatness. All of these arrangements facilitate working together in an informal manner, in contrast to the bureaucratic standoffishness that too much paper tends to encourage.

This intensity of communication, the working of problems together, has another effect—to optimize the entire system, rather than to suboptimize on the subsystems. I can illustrate this effect by reference to the engineering-manufacturing interface. American designers and manufacturing experts in modern times have been separate. As an engineer, you design and optimize and then throw it over the fence. The manufacturing experts catch it, but then they have to redesign it to make it manufacturable. This is an example of subsystem optimization.

Now American industry wants to do what the Japanese have been doing and are giving it a new name—design for manufacturability, that is, designing not only in terms of function, but in terms of making the product easier to manufacture. This is an attempt to optimize the total system. The Japanese methods of working together better enable them to apply total systems optimization to their software projects.

Positive characteristics

Three characteristics of the Japanese appear to have particular relevance to the ability to do software well: the sharp separation of planning and implementation, the willingness to do everything well even if it is boring, and the strong motivation to cooperate with each other.

Indeed, it seems that a Japanese either plans or implements. During planning no time and effort are spared to put all the details together. During implementation the execution of the plan is almost mindless. In contrast, the Western way mixes implementation with improvised innovation—whether the latter is desirable or whether it might undo earlier agreements with other participants.

I was startled to see the energy and effort put to a task regardless of whether it was intellectually challenging or not, novel or repetitious, short- or long-term. The intensity applied to the job seemed always the same.

The Japanese always tries to optimize a larger entity than his individual self: the target is the company, the project, the team. Pride of accomplishment is derived from the success of a product, which is the fruit of team effort. This is in contrast to the large management effort and psychic energy demanded and spent on compromising conflicts triggered by jealousy and often self-destructing rivalries in the West.

As people participate in both the planning phase and the implementation phase, they acquire much more complex knowledge. The knowledge exists in their skulls, not on pieces of paper. As you move the people around, they carry the technology from one place to another.

There are at least two paradoxes in Japan. One is between tradition and novelty. Tradition is strong in Japan, yet it accepts novelty in technology.

The other is between integration and specialization. The Japanese software factories integrate all the phases of development from requirements to testing, perhaps even maintenance, but they are also specialized in another dimension —Toshiba in process control software, Hitachi in the Cobol world and data processing, and NEC in communications software. In their areas of specialization they are able to make much use of reusability.

All of these characteristics seem to have a positive effect on the improvement of software productivity and quality. Development is more predictable if it is well planned, if detailed work is not skipped over, and if activities are well coordinated in large projects.

A moving target

Yet I am less afraid of the Japanese competition now than before my assignment in Japan. There is a dark side to many of these characteristics. There are instances in Japanese history, which I won't go into, in which the whole country has been led into disaster by going too much on inertia. They have started on a course of action and become so obsessed by the beauty of this well-planned activity that they have continued to support it, even beyond the point where it became obvious that it was not working.

Similarly, it is not always possible to plan a software development perfectly. In any case one must not follow even the best plan into failure.

Perhaps as a by-product of the well-laid-plan way of thinking, the Japanese seem to have difficulty in handling exception cases, those that are not in the plan, and in coping with changes, technological or otherwise, that come up after the plan is set. It is nice to follow a good plan; you do not have to constantly reorganize and disrupt the work. Still, there are times when it is nice to be able to apply a fix on the fly that is outside the plan.

Indeed, the best counterstrategy for Japan's competitors may be to become a moving target. The moving target—the product of flexibility—is very difficult to follow, particularly if you are already following a very good plan. ☐

Laszlo A. Belady is vice president and director of software technology at Microelectronics and Computer-technology Corporation in Austin. A software expert with a quarter-century of experience, he lived and worked in six countries, predominantly in the United States. During 23 years with IBM he established software research in the Japan Science Institute (1983-84), was responsible for software technology at corporate headquarters (1981-82), did groundbreaking research in virtual systems, and in the seventies led the software engineering effort at the Yorktown Research Center (1961-81). One of his 40 papers became Citation Classic in 1983 as the most referenced article published in the field of software in two decades. He is recipient of many IBM and IEEE awards and of nine US patents. He was chairman of national and international conferences and editor-in-chief of the *IEEE Transactions on Software Engineering* (1980-83). A frequent lecturer worldwide, he was visiting professor at UC Berkeley, London University, University of Texas, University of Minnesota, and NY Poly. A graduate of the Technical University of Budapest (Aeronautical Engineering—honors), he also worked as design engineer in Hungary (Air Force), West Germany (Ford), and France (Dassault).

Readers may write to the author at Microelectronics and Computer-technology Corporation, 9430 Research Blvd., Austin, TX 78759.

Special Feature

A Visual Approach to Browsing in a Database Environment

James A. Larson, Honeywell Computer Sciences Center

The operations of structuring, filtering, panning, and zooming can be used to browse the contents of a database.

A visual approach to browsing in a database environment is one way for users to access database contents easily and conveniently. People relate better to visual representations of the logical database than to other representations.

Users perform four basic operations when browsing a database: *structuring,* choosing a structure of the objects to be examined; *filtering,* selecting instances of the objects to be examined; *panning,* examining neighboring object instances; and *zooming,* determining the level of detail for examining object instances. My approach consists of four very flexible steps corresponding to each of the four basic operations. Each step uses visual representations to present information to the users. By manipulating these visual representations, users can structure, filter, pan, and zoom. Users may revise decisions and choices made during any of the four steps.

Most database management systems require users to formulate a complex specification describing the data that they wish to access. In order to do this, they must be familiar with the logical structure of the database; they must know the names of the types of objects to be accessed and how these objects are related. They must also be able to specify which occurrences of the objects are to be accessed by describing the criteria accessed occurrences must satisfy. This is difficult for many users because of their unfamiliarity with the syntax used to formulate requests, or because they have only a vague notion of what data they desire to access.

Users who have only a vague notion of the data to be accessed need to browse through the database. By browsing through the bookshelves of a library, it may be possible to locate one or more books of interest. By browsing through the pages of these books (perhaps making use of indices and tables of contents) readers may be able to locate the desired information that they could not previously characterize or describe. Database users have a similar need to browse through information in a database. A technique is needed for users of database management systems to locate information without having to specify an exact description of the information or where it is stored in the database. Such a technique is useful when users initially have only a vague idea of what they desire, but feel confident that they will recognize the desired information when they see it.

Browsing operations

To understand the four basic browsing operations, consider a paradigm of a hardware store customer desiring to purchase

0018-9162/86/0600-0062$01.00 © 1986 IEEE

some lightbulbs. The floor plan of the hardware store is arranged so that customers can locate desired merchandise by themselves. The customer quickly proceeds to the lightbulb counter, bypassing other merchandise not of interest. The customer scans the several brands and styles of lightbulbs on the counter and closely examines individual lightbulbs as he or she selects the desired lightbulbs for purchase. This paradigm illustrates the four browsing operations of structuring, filtering, panning, and zooming.

Structuring. Choose a structure for the objects to be examined. Often this is done by someone other than the person doing the browsing. For example, the hardware store manager would arrange the floor plan of the store in such a fashion that customers can easily browse through the items for sale. A few database management systems interfaces permit the user to structure objects to be examined. For example, GUIDE[1] permits the user to identify an object type to examine from a schema. The user may also examine object types that are related to the identified object type. The user may examine object types that are indirectly related to the identified object type via a chain of related object types. The result is a subschema that the user may use to formulate queries. Many database management systems permit only the database administrator to structure subschemas or views of the database.

Filtering. Select the instances of the objects to be examined. For example, if a customer is interested in purchasing lightbulbs, he or she will select one or more lightbulbs from the lightbulb counter containing lightbulbs of various wattages and brands. Some researchers[2] refer to this operation as *probing*.

Many degrees of filtering are possible:

No filtering. The user may not restrict the object instances to be displayed by specifying a query. SDMS [3] falls into this category.

Manual query specification. Most systems permit this option.

Manual query modification. Some systems save a user's query and permit the user to perform surgery on the query, modifying it in various ways.

Assisted query modification. System utilities allow the user to combine two queries into a single query, or to use the results of one query as part of the formulation of another query.

Automatic query modification based on user feedback. A revised query is automatically generated based on the user's determination of the relevance and nonrelevance of instances displayed as a result of the original query.

Retraction. This is a process of expanding a query to retrieve possibly relevant object instances when the original query fails to return any object instance.[2]

Panning. Examine the object instances adjacent to an object instance. In my example, the customer may examine several lightbulbs at the same counter. Motro[2] calls this operation *navigation.* Most interfaces to database management systems permit the scrolling or paging[4] of object instances onto a video screen permitting the user to pan.

Zooming. Determine the level of detail for examining object instances. For example, the customer may examine the guarantee in fine print on the lightbulb container, or may examine the lightbulb for flaws. Zooming permits the user to see (or hide) a level of detail that previously was (was not) visible. The user can zoom in or out to view object instances at the level of detail desired. SDMS,[3] VIEW,[5] PDB,[6] LID,[7] and the interactive language of Lochovsky and Tsichritzis[8] all permit zooming. However, the above systems place different restrictions on the number of levels of detail the user can view. For example, PDB and LID permit the user to view at most two levels at a time; the Lochovsky and Tsichritzis's language permits the user to view three levels, while SDMS and VIEW permit the user to view an arbitrary number of levels.

The MPS[9] is a system that could be used as an interface to a wide variety of computerized systems, including a database management system. MPS includes components that support these four operations. The MPS "presentation surface builder" is used for structuring and filtering a two-dimensional presentation surface containing the representation of instances of objects to be examined. The "presentation surface effector" is used for panning and zooming much like SDMS.[3]

A visual approach to browsing

Users often have difficulty visualizing the logical structure of a database. One reason is that databases often have many instances of entities that may be related in many ways. A visual approach to browsing is desirable because many people tend to relate better to visual representations. In addition, users can understand entity relationships quickly and manipulate them easily.

To simplify the browsing of a database, a schema graph is used. A schema graph illustrates the logical structure of a database and hides the differences in syntax and data models of various database management systems. A *schema graph,* $S=(V, E)$ is a graph consisting of V, the set of vertices that represent entity types in the database, and E, the set of edges that represent relationships between entity types. In a CODASYL network database, V represents CODASYL record types and E represents CODASYL set types. In a relational database, V represents the set of tables and E represents relationships between tables as identified by common key column(s) of one table and the foreign key column(s) of another table.

In my approach to browsing, the user first chooses a portion of the schema that contains entity types in which the user has interest. The user then specifies restrictions to the instances of these entity types and finally examines the entity instances selected from the database. My approach to database browsing can be summarized by the following four steps, which correspond to the browsing operations outlined above.

Step 1. Structuring. Choose a subgraph of the schema graph that specifies a hierarchy of entity types in V. This subgraph defines a forest of trees, each tree consisting of several related entity instances from the database.

Step 2. Filtering. Thin the forest of trees defined by the subgraph. A query language is used to specify the desired tree instances from the database to be examined

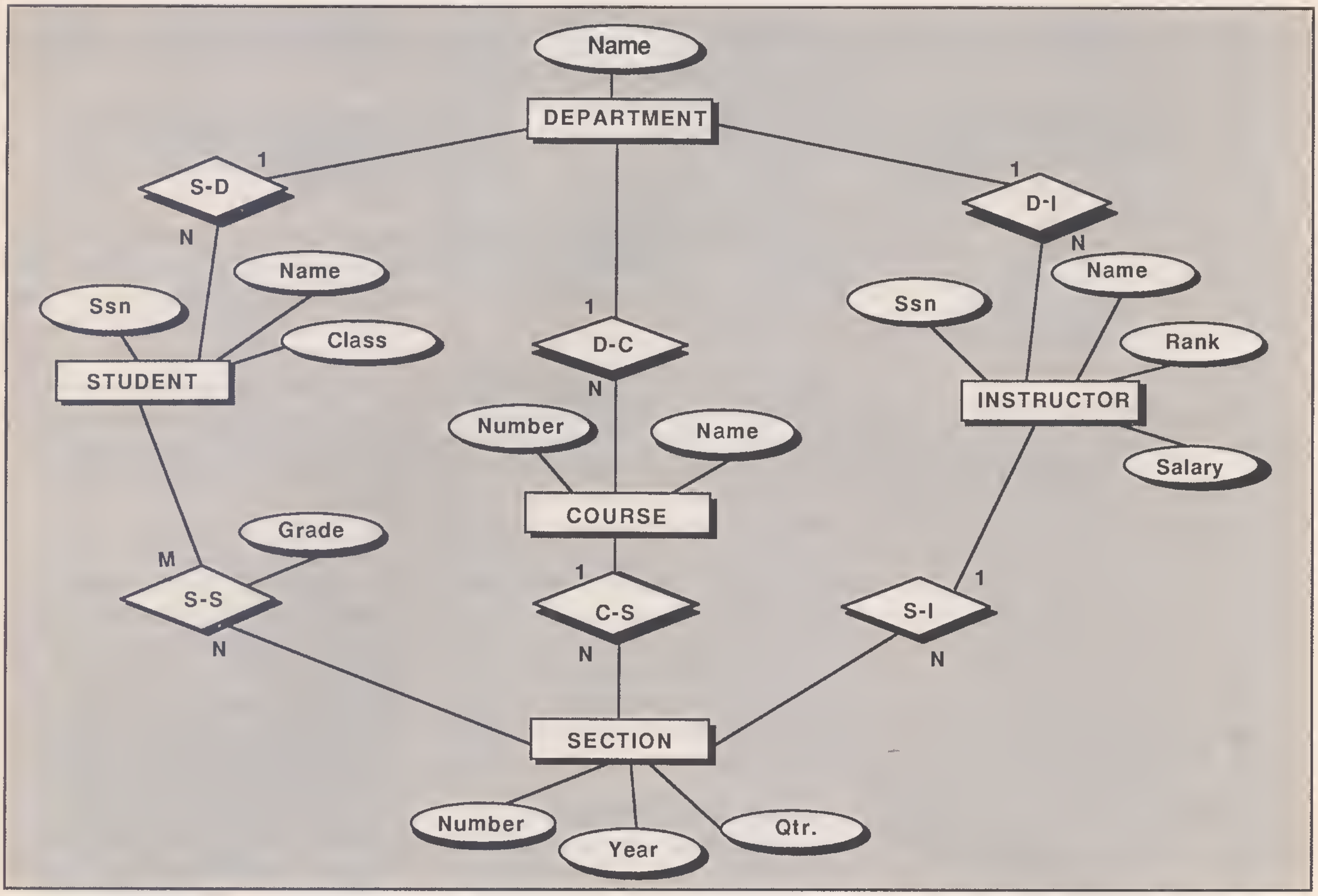


Figure 1. Graphical representation of schema.

in detail. The query language is a visual interface similar to Query-by- Example.[10]

Step 3. Panning. Display the selected entities in the trees of the thinned forest. The user views each tree as a series of nested documents on a video screen. Each document displays the attribute values of an entity instance from the database. The user may page and scroll through the documents.

Step 4. Zooming. Choose the entity types whose instances are to be displayed. The user may elect to display only the roots of the trees, only the leaves of the trees, only the entities at a specific level in the trees, or a combination of these.

Each of these steps makes use of visual representations to present information to the user. The user chooses options and specifies parameters by manipulating these visual representations. To better understand these steps I will describe the mechanisms that allow them to work.

Step 1: Structuring. In a graphical representation of a schema graph (Figure 1) based on Chen's Entity-Relationship data model,[11] rectangles represent entity types, diamonds represent relationships between entity types, and ovals represent attributes of entity types and relationships. The cardinalities of a relationship are indicated by numbers at the points of the diamond representing the relationship.

A form template (Figure 2) describes a hierarchical subgraph of the schema graph of Figure 1. The user chooses a subtree represented by a form template by following the six steps described later in this section.

A form template is a two-dimensional layout of one or more entity types. Each entity type is represented as a window on the screen. The nesting of windows corresponds to the hierarchical relationship of the entity types. When displayed on a video screen, a database entity is displayed in each window. Each window also contains "background" information that includes the window name and the names of attributes of the corresponding entity type. In Figure 2, the background information is that which is not underlined. Each window also contains example values. Example values are underlined in Figure 2. Attribute values from an entity instance replace these example values when entity instances are displayed. (Because attribute values are not underlined, they are not confused with example values.)

A visual interface[12] can guide the user through five steps to generate a form template that describes a subgraph of a schema graph.

Step 1a. A schema graph is automatically generated on the screen from information in a data dictionary. To generate this graph, the system reads adjacency information about entity types, relationships, and attributes that reside in the database and automatically derives the layout of the associated schema graph. This step relieves the user of the burden of remembering the exact contents and structure of the schema.[13,14]

Step 1b. The user examines the schema graph to view the structure of the database.

Step 1c. The user now trims off irrelevant entity types from the schema graph. The system supports two modes for doing this. The user can either specify the relevant entity types or specify the irrelevant entity types. When all relevant or irrelevant entity types have been identified, the system erases irrelevant entity types from the displayed graph. When the user has completed this stage, the displayed graph contains only entity types, relationships, and attributes that are to appear on the form template.

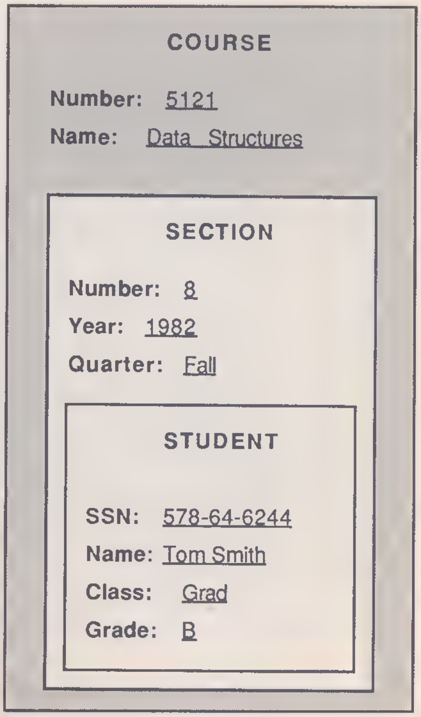
Step 1d. The schema graph must now be converted from a graph structure to a tree-structured, form-template view of the database. To aid in the conversion, the user identifies an entity type of primary interest. The designated entity type will become the root entity of the tree structure and will correspond to the top-level or outermost window in the form template.

Step 1e. The system automatically translates the schema graph into a nested form template. There are two stages necessary for this translation. The first stage[13] converts the schema graph into a tree of entity types. The second stage converts the entity tree into a form template. To convert the schema graph into a tree, the attributes of relationship type are migrated from the relationship downward (from the root) to the entity type below the relationship in the tree, and all relationship nodes are eliminated. In the hierarchical view, M:N and 1:N relationship types are viewed as 1:N, while N:1 and 1:1 relationship types are viewed as 1:1. Any two entity types in the hierarchical view with a 1:1 relationship type between them are merged to form a single entity type with the attributes of both entity types.

The tree is then automatically converted into a nested form template as follows:

1. The root of the tree becomes the top-level or outermost window of the nested form template.
2. The nesting structure of the form template windows is made to correspond to the tree of entity types.
3. Attribute names are positioned in the window corresponding to the appropriate entity type. Each attribute name appears on a separate line as background text.
4. An example value is a value from the domain of an attribute. It is displayed in the window as an example of a valid value for that attribute. An example value for each attribute is selected from a default list of example values for the attribute. Example values are placed on the same line as and to the right of the attribute name.
5. A name corresponding to the entity type name is placed as background text at the top of each window in the nested form template.
6. Appropriate information is stored in the data dictionary. This includes information about entities that were merged during the hierarchical conversion. This information is used when translating queries formulated using the form template into equivalent queries suitable for processing by the underlying database management system.

Users may manually change the layout of a window by repositioning the attribute names, example values, or entity type name within the window. Users may modify background information (attribute names and entity type name). The resulting form template describes a forest of trees embedded in the database graph. The form template may be used to thin this forest of trees.


COURSE

Number: 5121
Name: Data Structures

SECTION

Number: 8
Year: 1982
Quarter: Fall

STUDENT

SSN: 578-64-6244
Name: Tom Smith
Class: Grad
Grade: B

Figure 2. Form template.

An example. As an example, suppose the graph schema of Figure 1 is displayed on the screen. Suppose that the user wants a nested form template consisting of entity types COURSE, STUDENT, and SECTION. The user points to those three entity types and their relationships, S-S, C-S, on the screen. The system then erases the remaining entity types and relationships from the screen. The resulting subgraph is illustrated in Figure 3.

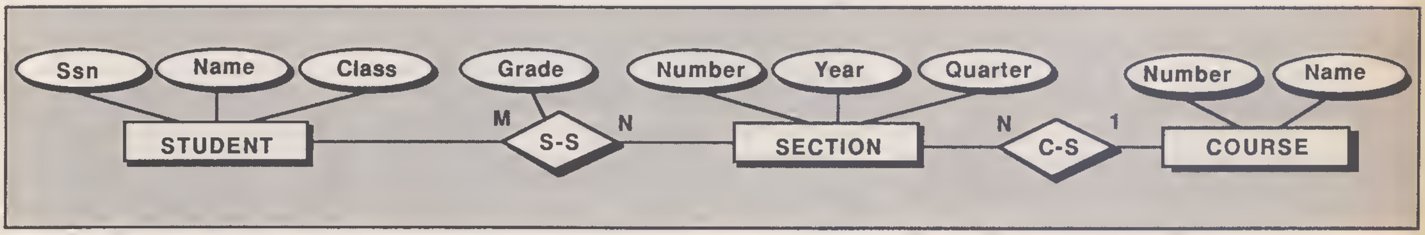


Figure 3. Schema subgraph.

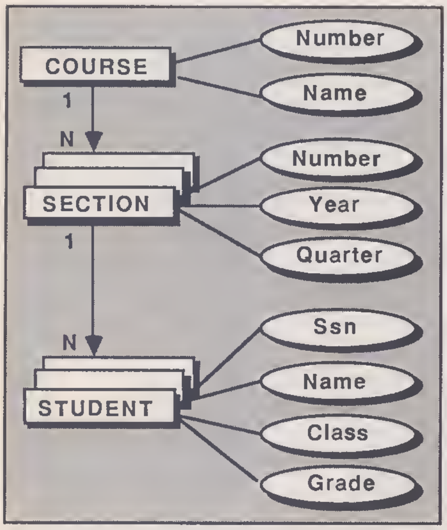


Figure 4. Schema hierarchy.

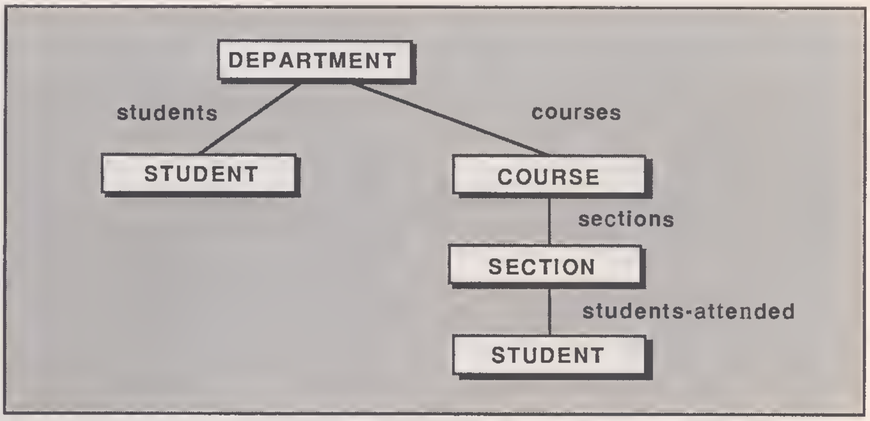


Figure 5. Converting a cycle to a hierarchy.

Now the user must choose a root entity type. Several choices can be made. Criteria[13] for choosing a root entity type include:

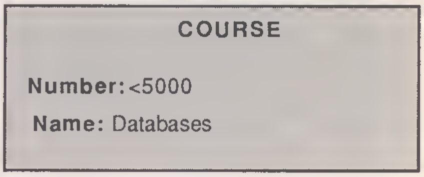
1. The entity type with the most attributes upon which conditions will be specified by users.
2. The entity type with the most attributes to be displayed in the nested form template.
3. The entity type that seems most natural for the types of information to be accessed. For example, if the user is most likely to browse between instances of STUDENT and only occasionally examine instances of SECTION or COURSE, then STUDENT should be selected as the root. On the other hand, if the user is more likely to browse between instances of COURSE, then COURSE should be selected as the root entity type.

Suppose the user chooses COURSE as the root. Relationship attributes are migrated to the entity types at the lower end (toward the leaves of the tree) of the relationship in the hierarchy. Thus, the Grade attribute is migrated from the S-S relationship to the STUDENT entity type, and gives the grade value of each student who took that section of a course. The subgraph of Figure 3 is converted into the hierarchy of Figure 4 and the corresponding form template of Figure 2.

Note that it is still possible to have a hierarchy even if the graph selected in step 4 has a cycle. In this case, two leaf entity sets are different. For example, if the user chooses DEPARTMENT, D-C, COURSE, C-S, SECTION, S-S, and STUDENT, S-D, then the user sees the hierarchy as shown in Figure 5. Here, the STUDENT related to DEPARTMENT directly shows the students who major in that department. The STUDENT related to COURSE shows students who completed some section of some course offered by the department.

Step 2: Filtering. The user thins the forest of trees to be viewed by specifying a pattern.[15] A *pattern* is a form template with zero, one, or more conditions. Figure 6 illustrates a pattern consisting of two conditions: (1) the attribute value for COURSE Name must be 'Databases' and (2) the attribute value for COURSE Number must be less than 5000. An instance of an entity must satisfy all conditions in the window if it is to be displayed to the user.

Patterns may be combined to form more complex patterns. The user may build a complex pattern by using pattern construction operators. Figure 7 illustrates several complex patterns that have been constructed using the pattern construction operators shown in Table 1. These pattern construction operations can be applied recursively to build patterns of arbitrary complexity.



COURSE

Number: <5000

Name: Databases

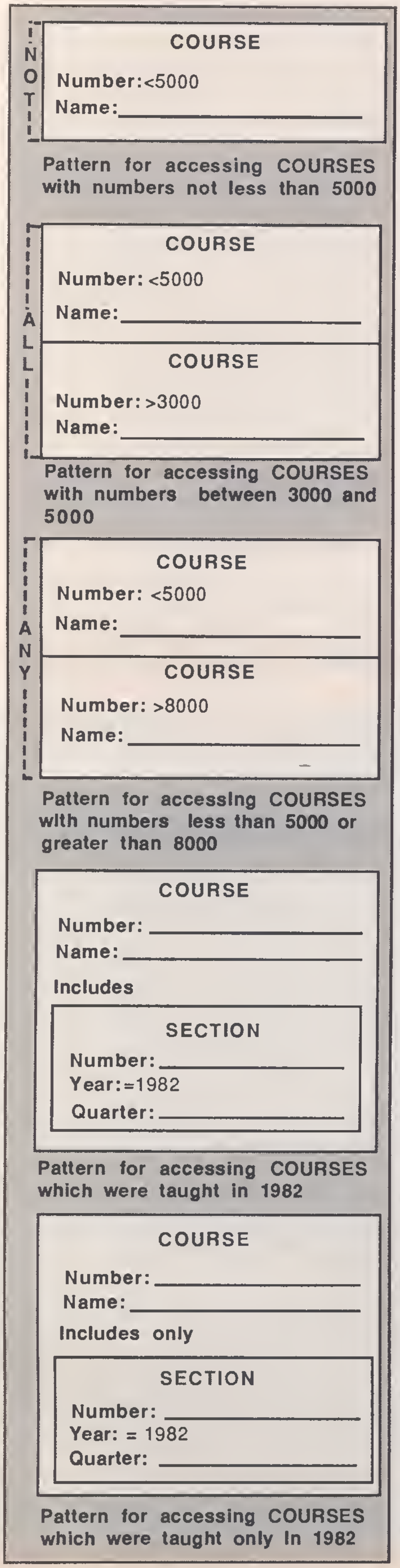
Figure 6. Example pattern.

Step 3: Panning. The user views the entity instances of each tree of the thinned forest as a series of nested documents. The entity instances of a tree are displayed in order sequence as a nested document (Figure 8). The user may page forward and backward between documents. By choosing a paging increment, the user may jump forward or backward a specified number of documents. The user may also display the first or last document and scroll within documents.

Step 4: Zooming. Each node of the tree constructed in Step 1 corresponds to a window in the form template that may be either open or closed. An *open* window displays all the attribute values of an entity instance. A *closed* window displays only the name of the entity instance. Users may open or close any window, depending upon if they wish the attribute values of entity instances of that entity type to be

Table 1. Pattern construction operators.

Pattern construction operator	Meaning	Restriction on use of this operator
Zero or more conditions	Select entity instances satisfying all conditions	Applies to exactly one window
NDT	Select entity instances NOT matching the pattern	Applies to exactly one pattern
ALL	Select entity instances matching ALL of the patterns	Applies to two or more patterns
ANY	Select entity instances matching ANY of the patterns	Applies to two or more patterns
INCLUDES	Select entity instances that have related entity Instances that INCLUDE an instance matching these patterns	Applies to one window nested within another window
INCLUDES ONLY	Select entity instances that have ONLY related entity instances matching these patterns	Applies to one window nested within another window



COURSE
Number:<5000
Name:

Pattern for accessing COURSES with numbers not less than 5000

COURSE
Number: <5000
Name:

COURSE
Number: >3000
Name:

Pattern for accessing COURSES with numbers between 3000 and 5000

COURSE
Number: <5000
Name:

COURSE
Number: >8000
Name:

Pattern for accessing COURSES with numbers less than 5000 or greater than 8000

COURSE
Number:
Name:
Includes

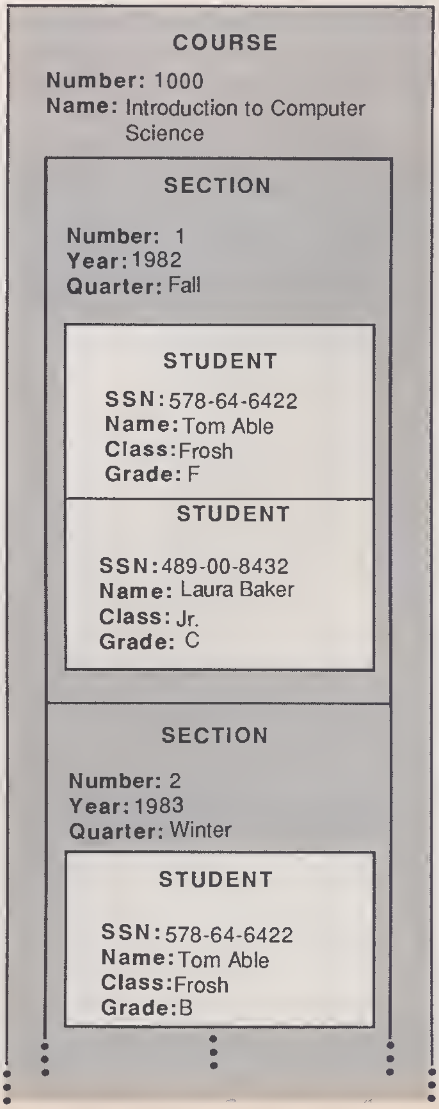
SECTION
Number:
Year:=1982
Quarter:

Pattern for accessing COURSES which were taught in 1982

COURSE
Number:
Name:
Includes only

SECTION
Number:
Year: = 1982
Quarter:

Pattern for accessing COURSES which were taught only In 1982



COURSE
Number: 1000
Name: Introduction to Computer Science

SECTION
Number: 1
Year: 1982
Quarter: Fall

STUDENT
SSN: 578-64-6422
Name: Tom Able
Class: Frosh
Grade: F

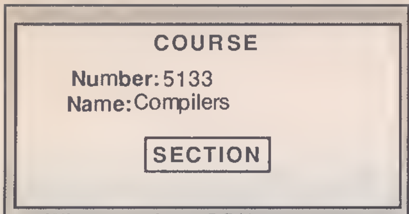
STUDENT
SSN: 489-00-8432
Name: Laura Baker
Class: Jr.
Grade: C

SECTION
Number: 2
Year: 1983
Quarter: Winter

STUDENT
SSN: 578-64-6422
Name: Tom Able
Class: Frosh
Grade: B

Figure 7. (Left) Example patterns.

Figure 8. Nested document.

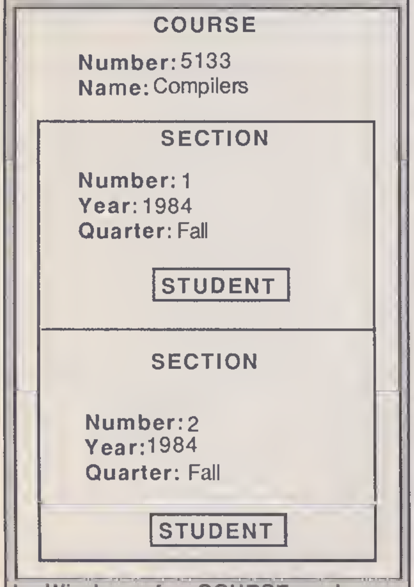


COURSE

Number: 5133
Name: Compilers

SECTION

a. Window for COURSE open



COURSE

Number: 5133
Name: Compilers

SECTION

Number: 1
Year: 1984
Quarter: Fall

STUDENT

SECTION

Number: 2
Year: 1984
Quarter: Fall

STUDENT

b. Windows for COURSE and SECTION open

Figure 9. Opening and closing windows.

visible when the selected entities are displayed. Opening and closing windows allow the user to control the level of detail in which the results are viewed. Figure 9a illustrates the Figure 2 form template with the windows for SECTION closed. Figure 9b illustrates the result of opening the SECTION.

Revising choices

A user may revise decisions and choices made during any of the four steps allowing for a great deal of flexibility. The user may always go back to step 1 and choose a new tree structure. The user may return to step 4 at any time to open (or close) any closed (or opened) windows. This changes the entity types that may be viewed. Within step 3, the user may view any of the selected entity instances by paging and scrolling between instances of entities. The remainder of this section discusses how the user can revise the pattern formulated in step 2 to thin the forest.

To change the way the forest of trees is thinned, the user may return to step 2. This capability is important because it encourages the user to thin the forest in an experimental fashion. In this way, the user does not have to worry about formulating a pattern correctly the very first time; a pattern that has been used to specify which entity instances can be viewed remains available for further modifications; and, the user does not have to rebuild the pattern from scratch each time it is modified.

The user might not correctly build the pattern in the first attempt for reasons including the following:

- The user is unclear about which entity instances to display.
- Particular entity instances may motivate the user to modify the initial pattern.
- The pattern may be too complex to formulate in one attempt.
- The user wants to view intermediate results.

Some existing approaches to browsing using graphics

FORAL LP,[1] Query By Example,[2] CUPID,[3] and GOING[4] are examples of pictorial graphical interfaces for database management systems. With these systems the user formulates a request using the pictorial interface, and then scrolls through the results. As with keyword-oriented query languages, the user can browse through only the results. The user must formulate a new request if the original request does not produce the desired results. Several efforts at providing more natural approaches to browsing through a database make use of graphics.

Lochovsky and Tsichritzis's interactive query language.[5] This language is designed for accessing several databases using a videotex system. The user sees a video screen divided into three basic windows. One window specifies the path taken by the user to arrive at the collection of named Information that can be examined. A second window of the screen is used for system feedback to the user. Finally, the third window is used for the specification and display of data. This window may display a menu consisting of two columns of named information. When the user selects a name in the left-hand column, that name is appended to the path name and a new menu is displayed showing some of the names previously in the right-hand column as names in the left-hand column. New names appear in the right-hand column. The user can thus browse as he or she specifies the path to the desired information. The third window of the screen may also display a template into which the user may directly enter names rather than select them from the menu. Thus users familiar with the structure of the databases may enter path information directly. The third window of the screen may also display a template into which users may enter one or more values to select specific pieces of information, to be displayed. However, the selectiveness of this facility is not as great as other query languages.

PDB, or Personal Data Base.[6] This displays the values of a single entity instance on the screen. The relationships in which the entity instance takes part is also displayed as a menu. The user selects one of these relationships and the system displays the values of the corresponding entity instance and a new menu. The user can thus navigate from entity instance to instance of different entity types in any desired order. LID, or Living In a Database,[7] is similar to PDB; however, LID uses a bit-mapped graphics terminal to display the attributes of an entity instance and entities that are related to it. To locate a specific entity instance, the user may either scroll through instances of the entity type or use a search function by filling values in a template. A graph structure is displayed that shows other entity instances that are related to the current entity instance. When the user moves the cursor to a related entity instance, the attributes of the related entity instance are displayed and a new relationship graph is drawn.

GUIDE, or Graphical User Interface for Database Exploration.[8] This displays a database schema in graphical form as relationship types. Queries can be expressed as traversal paths on this network. Partial queries can be formulated and represented graphically. Database retrieval results of any local query are available at any time. Partial queries can be linked together to form more complex queries and provide the bases for building queries in a piecemeal fashion. Parts

- The user makes a mistake when constructing the original pattern.

- Three approaches for pattern modification are explicit pattern modification, relevance feedback, and approximate matching.

Explicit pattern modification. The user can modify a pattern by explicitly modifying the conditions in any simple pattern, by creating or deleting patterns, and by changing the pattern construction operators of a complex pattern. The user may also move one or more entity instances into a pattern and use the pattern operators to combine the entity instances as patterns into more complex patterns.[16] This is especially useful because a particular database tree may motivate the user to modify the original pattern.

Relevance feedback. This is a technique for reformulating a pattern based on the initial retrieval. The aim is to use the revised pattern for a second retrieval that improves on the first. While explicit pattern modification permits the user to "perform surgery" directly on the pattern, relevance feedback performs this surgery automatically as follows:

1. The user formulates the initial (seed) pattern that is applied to the database.
2. The user examines some of the selected trees and determines if they are relevant or not relevant for his or her needs. Based on these relevance decisions, the original pattern is replaced by the conjunction of two patterns: the first describes the relevant documents and the second is the NOT of the pattern describing the nonrelevant trees. The revised pattern is applied to the remaining unexamined trees. This step is repeated until the user either examines all of the trees or determines that a sufficient number of relevant trees are examined for his or her needs.

Approximate matching. It may be desirable to access a database tree that "almost matches" a pattern (or a document being used as a pattern) for reasons including the following:

- The user wishes to view trees that could be viewed with only minor changes to a pattern without actually changing the pattern.
- The user is viewing a tree and is unable to formulate a pattern that will cause similar trees to be viewed.
- The user formulates a pattern that results in no trees and wishes to generalize the pattern by relaxing the conditions that the trees must satisfy so that more trees will be selected.
- The standard rules of a Boolean logic-based system may not be an ideal method of retrieving relevant trees. The operators "ALL," "ANY," and "NOT" (Boolean operators *and, or,* and *not*) have a strict interpretation. This may produce less than desirable results in certain cases. For example, in a pattern containing many subpatterns connected by an "ANY" operator, a tree satisfying all of the subpatterns would be treated the same as one satisfying only one of the subpatterns. Also, in a pat-

of the schema can be selectively made visible or invisible; this provides the basis of representing multiple levels of details of the schema.

SDMS, or Spatial Data Management System.[9] This is a human-machine interface that uses computer graphics to aid a user in interrogating a database. Objects in the system are represented graphically as icons positioned in a spatial framework consisting of a set of flat surfaces that can be viewed using a set of color, raster-scan displays. The user can create and examine data surfaces larger than the display screen, zooming in and out to control the level of detail displayed. The system allows the user to browse through a wide range of data by using a joystick. VIEW[10] extends the SDMS technique by enabling a variety of presentations of the data to be generated, without the need for painstakingly explicit instructions from the database administrator. Changes in the presentation of data include rearranging, adding, and deleting entities on the display; adding or deleting displayed attributes of entities; and altering the means of presentation of an entity or attribute. A further extension, MPS, or Multipurpose Presentation System,[11] combines database, graphics, and AI technologies. Both the user and the application system may operate on the data, describe image layouts, manipulate viewing windows, and build display images.

References

1. Michael E. Senko, "FORAL LP: Design and Implementation," *Proc. Fourth VLDB,* 1978, pp. 225-267.
2. M. Zloof, "Query By Example: A Data Base Language," *Proc. 1975 Nat'l Computer Conf.,* Anaheim, Calif., June 1975.
3. N. McDonald and M. Stonebeaker, "CUPID: A User Friendly Graphics Query Language," *Proc. 1975 ACM-PACIFIC,* San Francisco, Calif., April 1975.
4. Yosihisa Udagawa and Setsuo Ohsuga, "Novel Technique to Interact with Relational Databases by Using a Graphics Display," *J. Information Processing,* Vol. 5, No. 4, 1982, pp. 256-264.
5. F. H. Lochovsky and D. C. Tsichritzis, "An Interactive Query Language for External Data Bases," *Proc. Eighth VLDB,* Mexico City, Sept. 1982, pp. 11-21.
6. R. G. Cattell, "An Entity-Based Database User Interface," *Proc. ACM SIGMOD Conf. Management of Data,* May 1980, pp. 144-150.
7. Dennis Fogg, "Lessons from a 'Living in a Database' Graphical Query Interface," *Proc. ACM SIGMOD Conf. Management of Data,* 1984, pp. 100-106.
8. Harry K. T. Wong and Ivy Kuo, "GUIDE: Graphical User Interface for Database Exploration," *Proc. Eighth VLDB,* Mexico City, Sept. 1982, pp. 22-32.
9. C. F. Herot, "Spatial Management of Data," *ACM TODS,* Vol. 5, No. 4, pp. 493-513.
10. Gerald A. Wilson and Christopher F. Herot, "Semantics vs. Graphics—To Show or Not to Show," *Proc. Sixth VLDB,* Montreal, October 1-3, 1980, pp. 183-197.
11. Gerald A. Wilson et al., "The Multipurpose Presentation System," *Proc. Ninth VLDB,* Florence, Italy, Oct. 30-Nov. 2, 1983, pp. 56-69.

It is possible to retrieve trees by relaxing the interpretation of the Boolean operators.

tern containing "ALL'ed" subpatterns, a tree satisfying all but one of the subpatterns would be just the same as a tree satisfying none of the subpatterns.

It is possible to retrieve trees by relaxing the strict interpretation of the Boolean operators. One approach is to replace the ALL and ANY operators by a more general operator that requires an entity occurrence to satisfy a certain percentage (or a minimum number) of subpatterns. I have not pursued this approach because I feel it would be difficult for users to apply such a general operator. Unlike the next approach, this approach cannot rank the selected documents by the expected degree of relevance to the user's request.

Another approach to selecting trees that are "close" but don't quite match a pattern is to use a metric that has been specified for calculating the distance between a pattern and a tree. Using this metric, trees that are "close" to a pattern can be ranked according to their closeness to the pattern and presented to the user. Using relevant feedback, the initial query can be made more general. This has an effect similar to automatic retraction.[2]

Future work

Future work includes conceptual extension of the four browsing operations to support an interface to expert systems, application libraries, and graphical information.

Expert system interface. The browsing system is an easy-to-use interface to a database management system. A similar interface can also be used to access the capabilities of an expert system. Briefly, the schema graph would represent the capabilities of the expert system. Structuring corresponds to organizing the display of capabilities and selecting the desired functions. Filtering consists of specifying parameters values required to perform the selected functions. Panning corresponds to examining the results of similar problems. Zooming displays clarification about how the expert system derived the result.

Application library interface. Form templates can also be used to application software in a library. By filling in the input parameters for a form template, the user invokes the corresponding application program. Results of executing the application are displayed as output parameters on the same form template and may also be automatically entered as input parameters on other form templates. Browsing through form templates would be equivalent to viewing the results of automatically invoked applications, each application using information entered by the user or derived by previously invoked applications.

Graphical information. The current implementation of the browsing system will permit only alphanumeric text to be displayed on the screen. Techniques to support the following functions are needed to fully integrate graphics into a browsing system: create, modify, and save graphical images as part of the background information of a form template; store and access graphical images within a database and display and modify those images within a form template; translate information represented alphanumerically in the database into graphical images and icons for display within form templates; and modify the database information when the user manipulates the corresponding graphical image in the form template.

We have seen that browsing in a database environment consists of the four basic operations of structuring, filtering, panning, and zooming. An interactive, visual interface to database management systems that supports these four operations has been designed and is being implemented. In my approach, it is always possible to revise choices made in earlier operations. Future work includes extending the interface to support expert systems, application libraries, and graphical data. An experimental evaluation is needed to analyze the strengths and weaknesses of this approach to browsing.

New technology such as video text that allows individuals to access databases from their television sets at home can make many databases available to everyone. A visual approach to browsing will be needed by individuals with little training in traditional command languages so that they can quickly and easily access desired information.

Information that cannot be accessed by a user is useless. Browsing is a natural way for people to access information in a database. Laboratory systems incorporating the four main features of browsing (structuring, filtering, panning, and zooming) already exist. These features will find their way into existing and new database management systems, making accessing data easier for a wider range of database management system users. □

Acknowledgments

Tim Giebelhaus, Jennifer Wallick, John Nomura, and Jim vanDruten each designed and implemented portions of a prototype system for browsing. Many of the ideas for structuring are due to Ramez Elmasri.

References

1. Harry K. T. Wong and Ivy Kuo, "GUIDE: Graphical User Interface for Database Exploration," *Proc. Eighth VLDB,* Mexico City, Sept. 1982, pp. 22-32.
2. Amihai Motro, "Browsing in a Loosely Structured Database," *Proc. ACM-SIGMOD Conf. Management of Data,* June 18-21, 1984, pp. 197-207.
3. C. F. Herot, "Spatial Management of Data," *ACM TODS,* Vol. 5, No. 4, pp. 493-513.
4. Michael Stonebraker and Joseph Kalash, "Timber: A Sophisticated Relation Browser," *Proc. Eighth VLDB,* Mexico City, Sept. 1982, pp. 1-10.
5. Gerald A. Wilson and Christopher F. Herot, "Semantics vs. Graphics—To Show or Not to Show," *Proc. Sixth VLDB,* Montreal, October 1-3, 1980, pp. 183-197.

6. R. G. Cattell, "An Entity-Based Database User Interface," *Proc. ACM SIGMOD Conf. Management of Data,* May 1980, pp. 144-150.
7. Dennis Fogg, "Lessons from a 'Living in a Database' Graphical Query Interface," *Proc. ACM SIGMOD Conf. Management of Data,* 1984, pp. 100-106.
8. F. H. Lochovsky and D. C. Tsichritzis, "An Interactive Query Language for External Data Bases," *Proc. Eighth VLDB,* Mexico City, Sept. 1982, pp. 11-21.
9. Gerald A. Wilson et al., "The Multipurpose Presentation System," *Proc. Ninth VLDB,* Florence, Italy, Oct. 30-Nov. 2, 1983, pp. 56-69.
10. M. Zloof, "Query By Example: A Data Base Language," *Proc. 1975 Nat'l Computer Conf.,* Anaheim, Calif. June 1975.
11. P. P-S Chen, "The Entity-Relationship Model: Toward a Unified View of Data," *ACM TODS,* Vol. 1, No. 1, March 1976, pp. 9-36.
12. James A. Larson and Jennifer Wallick, "Tools for Forms Administrators," *IEEE Workshop on Languages for Automation,* New Orleans, Nov. 1984; also, *Languages for Automation,* Shi-Kuo Chang (ed.), Plenum Publ. Corp., New York, 1985.
13. R. A. Elmasri and J. A. Larson, "A User Friendly Interface for Specifying Hierarchical Queries on a ER Graph Database," *Fourth Int'l Conf. Entity-Relationship Approach,* Chicago, Oct. 28-30, 1985.
14. Kazuo Sugihara et al., "A Semantic Approach to Usability in Relational Database Systems," *Proc. IEEE Conf. Data Engineering,* Los Angeles, April 24-27, 1984, pp. 203-210.
15. James A. Larson, "The Forms Pattern Language," *Proc. IEEE Conf. Data Engineering,* Los Angeles, April 24-27, 1984, pp. 183-192.
16. Jim VanDruten, *Incremental Refinement of Queries in an Electronic Forms System,* Honeywell Computer Sciences Center Technical Report, Aug. 1984.

James A. Larson is a Senior Research Fellow at the Honeywell Computer Sciences Center in Minnesota, where he is involved in research in distributed databases. Previously he has been involved in the design of integrated automated office systems for Olivetti, Italy, and research in database computers for Sperry Univac. He has participated on the CODASYL Data Description Language Committee and the ANSI/SPARC Relational Database Task Group.

Readers may write to the author at Honeywell Computer Sciences Center, 1000 Boone Ave., North, Golden Valley, MN 55427.

Computer Scientist

Software Development

United Technologies Research Center, a recognized source for broad ranging research of the first rank has gained wide recognition for significant achievements both in basic and applied research. Here you'll discover a research environment tailored to your needs . . . top researchers . . . ample funding . . . outstanding back-up services . . . a scientifically-oriented management . . . encouragement and reward for publication of papers.

Expanded activities in Computer Science Research offer you an opportunity to build professional recognition in a wide spectrum of assignments. Projects span the area of Ada, Ada support environments, and formal approaches to software development. The diversity of the projects offer wider exposure and earlier recognition of your talents.

We are particularly interested in candidates who have a Master's or PhD degree in Computer Science with significant course work in software engineeering, data-base/knowledge-base management, operating systems, formal specification and design languages, and Artificial Intelligence. Research experience focused on Ada, Ada support environments, UNIX and knowledge-based software development techniques would be advantageous.

Find out more about the advantages that include excellent compensation and benefits. Send resume with salary history in confidence to: Dr. M.G. Kuhnly, United Technologies Research Center, Silver Lane, East Hartford, CT 06108.

UNITED TECHNOLOGIES RESEARCH CENTER

An equal opportunity employer • U.S. Citizenship required

The Interpress page and document description language

by Abhay Bhushan and Michael Plass
Xerox Corporation

Potential users of computer-aided documentation systems almost inevitably face the problem of taking a document created on any of a diverse set of tools (workstations, mainframes, word processors, and scanners, etc.) and printing it on any of a broad range of sharable printing devices (from low-speed desk top printers and low-resolution proof printers to high-speed document production systems and high-resolution phototypesetters). More often than not, output devices each require different formats to characterize a document's design, making it virtually impossible to achieve any consistency of appearance between versions of the document printed on different devices. They may also require unique interfaces with device-dependent control codes, protocols, and configuration requirements. Such demands effectively prevent the user from exploiting the full capabilities of advanced printing technology. The Interpress page description language addresses this problem with a device-independent interface and a format description methodology that make it possible to efficiently employ a full array of output resources in a manner that is transparent to the user.

Toward device independence

Computer-driven raster printers are inherently expressive devices, capable of printing any imaginable combination of text, graphics, and pictures simply by arranging the appropriate pattern of black (or colored) dots. Unlike character- or line-oriented devices, the range of images they can print is limited more by the expressive power of the interface with the document creation tool than by their own capabilities.

Because raster printers are driven by computer software and do not use device-specific control codes, it is possible to have a universal interface or interchange standard in which any document may be represented, and that can drive any raster printer, independent of its resolution (expressed in dots per inch) and other device characteristics. Interpress is designed as such a device-independent standard. Since Interpress is so conceptually different from conventional printer interfaces, it is generally referred to as a page and document description language. The Interpress printing architecture is a software architecture that represents a unified document description scheme for printing in diverse system environments, including stand-alone computers, data processing centers, publishing and office work groups, and large, interconnected networks.

When work first began on Interpress over a decade ago at Xerox Palo Alto Research Center, the original goal was to design a system that permitted the same format to be used for editable (revisable form) as well as printable (final form) documents; however, this goal was soon seen as impractical and was discarded in favor of separate interchange schemes for revisable and final form documents. Interpress emerged as a language for final form document representation. The language definition was made publicly available in 1984 (an earlier version of the current Interpress 3.0), and Xerox is committed to keeping it an open system.

The Interpress approach

In one possible document description scheme, the document creation device would send to the raster printer a facsimile picture of the intended output, presented directly in that printer's raster format. Such a scheme, though conceptually simple, has many disadvantages. First, raster data, which amounts to a map of every point that the printer must address (a *bitmap*), takes up an enormous amount of storage space (millions of bits per page, even when compressed). This increases not only storage costs but also transmission time and costs. Further, if the page contains text, the computer program generating it must have access to the raster image of all of the fonts for all printers that might be used; such a requirement is quite impractical. It would also be difficult to transform the raster images—to rotate, scale, or move them to fit a particular space or to achieve a desired effect. Finally, the raster format is based on the resolution of the printing device and is therefore not device-independent.

By contrast with this pictorial approach, Interpress represents a page not as a set of points for the printer to address but as a series of instructions analogous to a computer program. This technique permits the user to print substantially more complex documents than is possible with the static format specifications of character printers, and to do so with greater efficiency than is possible with conventional raster data.

Figure 1 illustrates this approach to representing the letter "A." The pictorial representation takes several thousand bits at typical printer resolutions, the geometric representation takes several hundred bits, and the textual representation requires only eight bits once the font has been specified. While Interpress permits all three approaches, the textual approach is preferred in most applications. The geometric approach of outline characters is useful when unique character sizes are desired or when the characters must be rotated at unusual angles.

The program that drives the printing machine to produce the finished output document is called an Interpress master, written in the Interpress programming language. Although most masters consist only of simple statements, such as text, rectangles, and line segments, the full

The Workstation is a forum for short articles on current research and developments in the computing sciences from industry, universities, and government. Submit articles of between 1000-3000 words in length to **Michael C. Mulder, Director, Applied Research, University of Portland, Portland, OR 97203.**

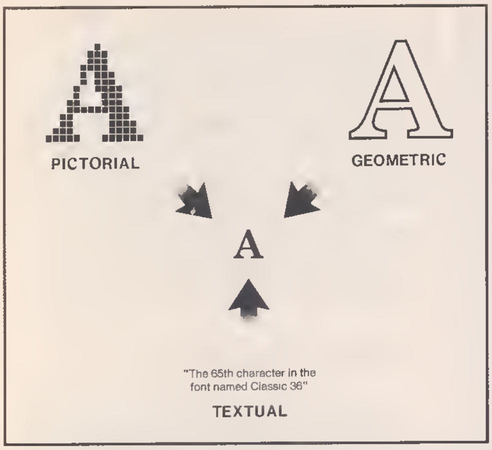

PICTORIAL
GEOMETRIC
A
"The 65th character in the font named Classic 36"
TEXTUAL

Figure 1. Three ways to print the letter A. (All figures in this article were produced on a laser typesetter at 1200 dots-per-inch resolution.)

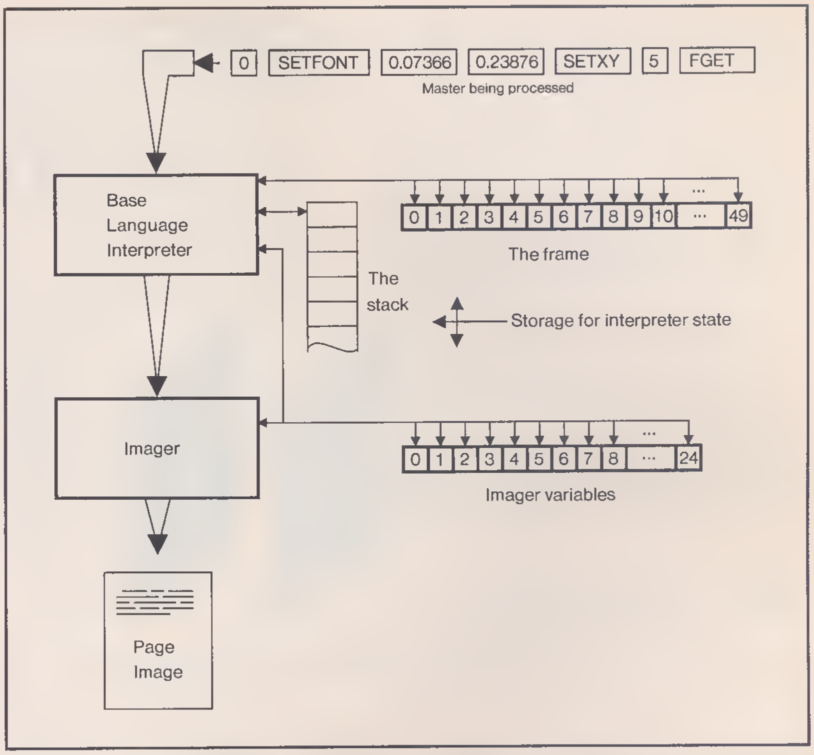


Figure 2. The structure of an Interpress printer.

power of the programming language is available for complex applications. Programming is useful if the master is to adapt to the various properties of the printing device (such as page size, order of page printing, color or black and white, etc.) and to change Interpress masters in complex ways.

Though a key task, describing pages is not enough. What is communicated to the printer usually is a document, which is a collection of one or more pages produced in a specific order and intended to have a specific relationship with one another in the document's final form. A document description language therefore must be able to specify how pages are put together. The Interpress design includes a set of printing instructions that enables the user to control the printing of documents—to invoke two-sided printing, for example, or a special finishing such as stapling. Printing instructions also provide information necessary for multi-user environments (the document's name, author, etc.) and enable the declaration of resources required for the the document to be printed (e.g., additional files, fonts, and font sizes).

The Interpress language

Like all programming languages, Interpress has both syntax and semantics. The semantics of the language define how the various operators behave when they are executed by the printer; the syntax of the language defines how the calls to those operators are coded in a master. Since Interpress masters are intended to be created and interpreted by software and not by people, the language syntax was designed to make it easy for computers to produce and interpret, without concern for human readability. As such, Interpress commands are normally encoded in a binary format designed for compactness and decoding ease. For debugging purposes, utility programs translate the binary encoding to and from a human-readable text representation; this "written encoding" is used in the examples in this article.

The software within a printer interprets (executes) an Interpress master to print a document. During that execution, the printed document is built up one page at a time. When a page of a master is printed, an interpreter "executes" the code that constitutes the page description, much as a machine executes a program. The state of this virtual machine includes a set of 50 "registers" called the *frame*, a (potentially large) *stack*, and a set of special *imaging variables*. This computational environment is shown in Figure 2.

The elements stored in the frame and stack are Interpress *values*, which may be of type *number* (integer or real), *identifier* (similar to atoms in other languages), *vector* (a packaged sequence of values), *body* (a packaged piece of Interpress code), or *operator* (an Interpress program that can be executed). Character codes are represented by integers, and character strings by vectors of integers; there are compact encodings available for this kind of vector to keep the master compact. Additionally, there are special types that are constructed and used only by imaging operators, such as *color*, *transformation*, *pixelArray*, *font*, *trajectory*, and *outline*.

Interpress uses a postfix execution model: the occurrence of a literal in the instruction stream simply causes the corresponding value to be pushed on the stack, and an operator may pop some number of parameters off the stack and push some number of results. There are provisions for marking the stack to ensure that defined procedures (*composed operators* in Interpress parlance) pop and push the expected number of items. There is also a way of saving and restoring some or all of the imaging variables around the invocation of an operator. Furthermore, composed operators do not read or modify the caller's frame; they have their own frame, initialized when they were created and the same at each call. Thus the side effects of composed operators are well controlled.

The imaging model

Of course the most important side effects are those that change the image on the page. The software maintains a page

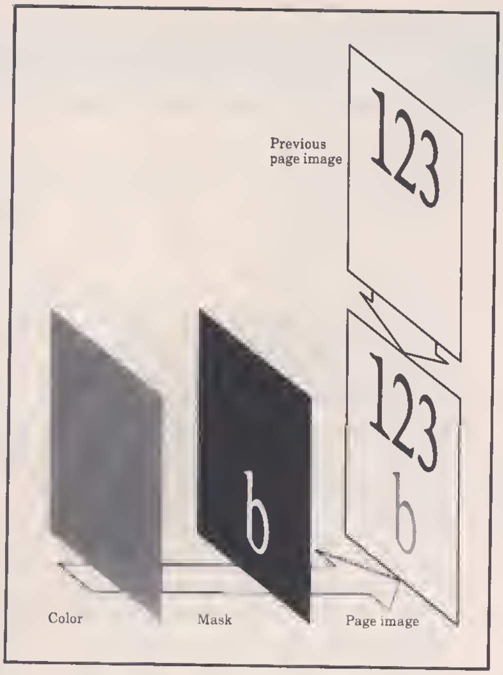


Figure 3. An imaging model, showing the application of a mask operator.

image, which is altered by the imaging operator as the page is built. A complex page is made by starting with a blank page image and making a sequence of simple changes to it. Interpress is defined so that the partially built page image cannot affect the execution of the master; this makes Interpress useful for expressing images destined for non-raster devices such as pen-plotters and certain phototypesetters. All of the changes to the page image are made according to the Interpress imaging model illustrated in Figure 3: a color is pushed through a mask onto the page image, covering up (or perhaps altering) what was there before. In Figure 3, the color (represented by the parallelogram) is set with <0.5 SETGRAY>, and the "b" on the page image is printed using the filled outline mask defining the character's shape.

Colors

Interpress uses the term "color" to designate a concept that is more general than what we mean in our day-to-day use of the term. An Interpress color may be simply black or white, or a shade of gray, or a "real" color like red or blue, or a generic color like "highlight," which just means something other than black; these are all examples of *constant colors*. Constant colors may be specified by means of the SETGRAY operator (for simple grays), or from the printer's environment (by supplying a name to the FINDCOLOR operator), or via a *color model*, which is simply an operator that accepts a vector of numeric parameters on the stack and returns a color on the stack. (Color models are normally obtained from the environment by means of the operators FINDCOLORMODEL or FINDCOLORMODELOPERATOR.)

The other kind of Interpress color is a *sampled color*, consisting of three parts: a rectangular array of pixel descriptions, the specification (color model) for how the pixels are to be interpreted, and a transformation to specify how the pixels of the sampled color are to correspond to the pixels of the page image. The sampled color conceptually tiles the whole page, so it may be used to produce textures and wallpaper-like effects, as well as full-color continuous-tone images. The simplest kind of sampled color consists of a one-bit-per-pixel array, with 1 denoting black and 0 denoting either white or clear; this is known as a *sampled black*. Interpress allows various types of compression in the encoding of the arrays of pixels; the ones that are currently defined in the system's Raster Encoding Standard are expressly for the one-bit-per-pixel case.

Masks

The other half of the imaging model is the notion of masks. A mask is simply a two-dimensional shape used as a stencil in the application of color to the page image; it specifies the portion of the page image to be colored. A mask may be a rectangle, a stroke of a specified width along polygonal or curved trajectories, a dashed or dotted stroke, an area bounded by a set of trajectories, a bitmap (possibly compressed) at any resolution, or a string of text in a particular font. Any of these masks may be used with any color. The current color is one of the imaging variables, as is the current font and the current transformation (more about these later); the different kinds of masks have different operators.

A simple example

To illustrate some of the basic features of Interpress, consider the example in Figure 4. This master prints two pages; lines 1 through 7 specify the first page (containing a rectangular box made up of four strokes) and lines 8 through 13 specify the second (containing the text "Print this"). Lines 0 and 14 constitute the skeleton that brackets the page bodies (the empty brackets in line 0 are the preamble, which can set up the initial frame used for each page body; thus fonts, for example, must be declared only once rather than in each page). If the page brackets on lines 7 and 8 were eliminated, only a single page would be printed with "Print this" inside the rectangular box, as shown on the right in Figure 4.

The operations that actually change the page images occur in lines 3 through 6 (MASKVECTOR) and line 12 (SHOW); the other lines set up the necessary imager variables (strokeWidth for the MASKVECTOR, font and current position for SHOW). All the dimensions in this example are expressed in meters, which is the unit of the initial coordinate system of Interpress; the origin is in the lower left corner of the page, with *x* increasing to the left and *y* increasing towards the top of the page.

Typographic printing

Typography is the art of designing and placing letterforms to create a legible and pleasing effect. The quality of typography largely depends upon the availability of character sets and fonts. The Interpress printing architecture includes the multilingual Xerox Character Code set and the Font Interchange set. Interpress also allows other fonts and character codes to be used and intermixed freely in a master. For example, the ASCII formatting characters such as carriage return and tab are not recognized by the system because there is no accurate way to interpret what these characters should do. Formatting operations are achieved by positioning commands such as SETXY. Interpress also does not generate ligatures automatically. Ligatures must have a separate character code and representation (provided in the Xerox character code set). This reflects an important principle of Interpress design: all decisions about presentation and formatting should be made by the creator, not the printer. This principle ensures that documents are printed accurately and have uniform appearance on different output devices.

Further facilities include letterform definitions expressed as character operators; positioning operators used to control the position of the letters; geometric transformations to scale, rotate, and translate a letterform so that it can appear in arbitrary size, rotation, and position on a page; and additional graphical operators to define underlines, strikethroughs, and the like. Positioning may be either absolute (with respect to the page) or relative (with respect to some other coordinate system such as a box within a page). To assure correct justification and margin alignment, the system provides a CORRECT operator for spacing adjustment. It also defines a flexible way to achieve kerning (alteration in the intercharacter spacing of pairs of characters) to achieve a better appearance.

Fonts may be stored at the printer in outline or bitmap form, or they may be communicated as part of an Interpress master. The bitmap fonts generally are fine-tuned for the characteristics of a specific printer and represent high typographic quality. Outline fonts represent greater versatility since they can be easily scaled and rotated to provide printing in any size and at any angle.

Graphics printing

In some sense, the term "graphics" applies to everything that goes on the page; here, though, the term applies to elements other than black text at normal sizes and orientations and the rectangles that normally appear with such text.

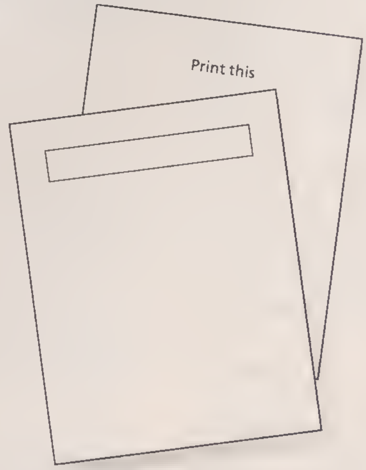
Rectangles are very simple to specify: the operator MASKRECTANGLE expects the four variables x, y, w, and h on the stack (x,y coordinates of one corner, the width, and the height). Single line segments, as we have seen, can be drawn by supplying the two endpoints (four numbers x_1, y_1, x_2, y_2) to the MASKVECTOR operator; Interpress allows control of the stroke width and the style of the end caps (square, butt, or circular). More complicated shapes are specified by building trajectories, which represent a sequence of straight or curved segments connected end-to-end. A trajectory is constructed segment-by-segment, using a pen-plotter analogy: the MOVETO operator takes the coordinates of a point (two numbers x_0, y_0) and returns a single-point trajectory. The other trajectory-building operators all take a trajectory followed by the appropriate number of coordinates and parameters, and return a new trajectory. A trajectory is an Interpress value and may be copied and saved for reuse; the most common pattern, though, is to construct it on the stack and use it right away. The other trajectory-building operators are:

x_1 y_1 **LINETO** straight line segment from (x_0,y_0) to (x_2,y_2);
x_1 y_1 x_2 y_2 r **CONICTO** conic segment —part of circle, ellipse, parabola, or hyperbola;
x_1 y_1 x_2 y_2 **ARCTO** circular arc passing from (x_0,y_0) through (x_1,y_1) to (x_2,y_2); and
x_1 y_1 x_2 y_2 x_3 y_3 **CURVETO** parametric cubic curve ending at (x_3,y_3).

Once a trajectory is built, it can be supplied to the MASKSTROKE operator to draw a constant-width stroke; the master may specify whether the joints between the segments should be mitered, beveled, or rounded. For a dashed or dotted stroke, the trajectory and a dash specification is provided to the operator MASKDASHEDSTROKE; fancy combinations of dashed and dotted strokes can be obtained by using the same trajectory with several different sets of dash specifications. MASKSTROKECLOSED draws a stroke with the segment closed by joining its two endpoints (with a line

```
Line
Number
(ref. only)  Interpress Master                                        Comments (for reference only)

  --0--    BEGIN { }                                                 part of the "skeleton"
  --1--    {                                                         beginning of the first page body
  --2--    0.001 15 ISET                                             set imager variable 15
                                                                        (strokewidth) to 0.001 meter
  --3--    0.0254 0.2286  0.0254 0.254   MASKVECTOR
  --4--    0.1905 0.2286  0.1905 0.254   MASKVECTOR
  --5--    0.0254 0.2286  0.1905 0.2286 MASKVECTOR
  --6--    0.0254 0.254   0.1905 0.254   MASKVECTOR
  --7--    }                                                         end of the first page body
  --8--    {                                                         beginning of second page body
                                                                     line 9 defines font and saves it in
                                                                        frame element 0
  --9--    [xerox xc1-1-1 modern] FINOFONT 0.0127 SCALE MOOIFYFONT 0 FSET
  --10--   0 SETFONT                                                 sets the "current font"
  --11--   0.08128 0.23622 SETXY                                     sets the "current postition"
  --12--   <Print this> SHOW                                         place "Print this" at current
                                                                        position in current font
  --13--   }                                                         end of second page body
  --14--   END                                                       end of master (more of the
                                                                        "skeleton")
```

Two pages images produced by this Interpress master.



One page image produced by this Interpress master without lines 7 and 8.

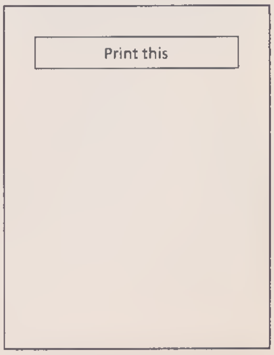
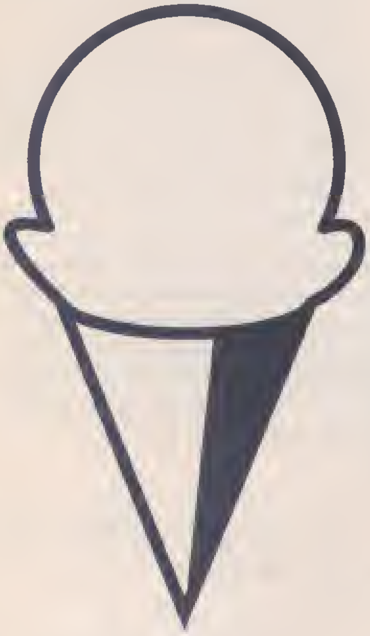
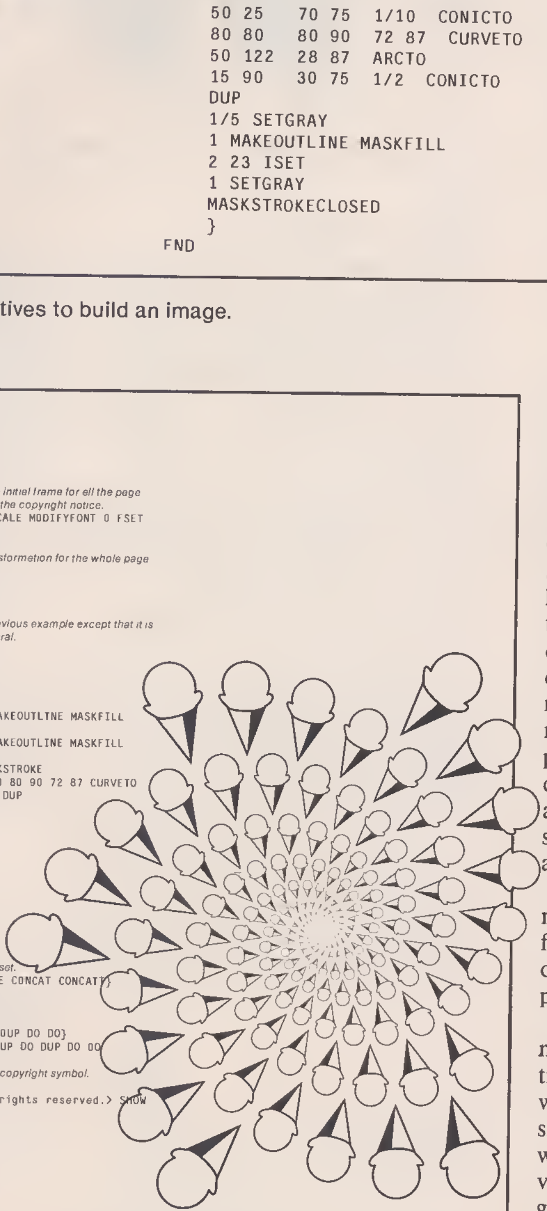


Figure 4. A sample Interpress master.



```
Interpress Master                                    Comments (for reference only)

BEGIN { }  {                                         will use overlapping shapes,
    1 5 ISET                                         so set priorityImportant to TRUE
    2/1000 SCALE CONCATT                             use 2 millimeter units

                                                     The Cone
    3/5 SETGRAY                                      darker gray side of cone
    70 75 MOVETO 50 25 LINETO 55 75 LINETO           a triangular trajectory
    1 MAKEOUTLINE MASKFILL                           fill the darker triangle

    3/10 SETGRAY                                     light side of cone is similar
    55 75 MOVETO 50 25 LINETO 30 75 LINETO
    1 MAKEOUTLINE MASKFILL

    2 16 ISET                                        round ends on the stroke
    0 23 ISET                                        miter to get a sharp point
    2 15 ISET                                        stroke will be 2 units wide
    1 SETGRAY                                        set current color to black
    70 75 MOVETO 50 25 LINETO 30 75 LINETO
    MASKSTROKE                                       line around cone bottom

                                                     The Ice Cream
    30 75    MOVETO
    50 25    70 75   1/10   CONICTO                  elliptical piece at bottom
    80 80    80 90   72 87   CURVETO                 parametric cubic at right
    50 122   28 87   ARCTO                           circular arc at the top
    15 90    30 75   1/2   CONICTO                   parabolic piece at left
    DUP                                              make a copy of trajectory
    1/5 SETGRAY                                      light gray for ice cream
    1 MAKEOUTLINE MASKFILL                           fill in the area
    2 23 ISET                                        use rounded joints for outline
    1 SETGRAY                                        set current color to black
    MASKSTROKECLOSED                                 draw stroke outlining area
    }
END
```

Figure 5. The use of graphics primitives to build an image.



```
BEGIN
{
  This is the preamble, the frame constructed here is used as the initial frame for all the page
   bodies (only one in this example). Here we just put a font for the copyright notice.
    Xerox xcT-1-1 Modern 3 MAKEVEC FINDFONT 10 SCALE MODIFYFONT 0 FSET
  }
{
  Start of the page body; set priorityImportant and provide a transformation for the whole page
    1 5 ISET
    1 SETGRAY
    10795/100000 13970/100000 TRANSLATE CONCATT
    1/3000 SCALE CONCATT
  Make a composed operator for a single cone; this is like the previous example except that it is
   transformed to its proper place relative to the center of the spiral.
    MAKESIMPLECO {
      0 200 TRANSLATE CONCATT
      -22 ROTATE CONCATT
      -50 -25 TRANSLATE CONCATT
      3/5 SETGRAY
      70 75 MOVETO 50 25 LINETO 55 75 LINETO 1 MAKEOUTLINE MASKFILL
      3/10 SETGRAY
      55 75 MOVETO 50 25 LINETO 30 75 LINETO 1 MAKEOUTLINE MASKFILL
      2 16 ISET 0 23 ISET 2 15 ISET 1 SETGRAY
      70 75 MOVETO 50 25 LINETO 30 75 LINETO MASKSTROKE
      30 75 MOVETO 50 25 70 75 1/10 CONICTO 80 80 80 90 72 87 CURVETO
      50 122 28 87 ARCTO 15 90 30 75 1/2 CONICTO DUP
      1/5 SETGRAY 1 MAKEOUTLINE MASKFILL
      2 23 ISET 1 SETGRAY MASKSTROKECLOSED
      } 10 FSET
  Make a composed operator for a ring of six cones.
    MAKESIMPLECO {
      10 FGET DOSAVE  -60 ROTATE CONCATT
      10 FGET DOSAVE  -60 ROTATE CONCATT
      10 FGET DOSAVE  -60 ROTATE CONCATT
      10 FGET DOSAVE  -60 ROTATE CONCATT
      10 FGET DOSAVE  -60 ROTATE CONCATT
      10 FGET DOSAVE  -60 ROTATE CONCATT
      } 11 FSET
  Do the above, but with a scale and rotate to do the next smaller set.
    MAKESIMPLECO {11 FGET DO 9/10 SCALE -15 ROTATE CONCAT CONCATT}
      12 FSET
  Do that many times to draw the spiral
    DOSAVESIMPLEBODY {
      MAKESIMPLECO {12 FGET DUP DO DUP DO DUP DO DUP DO DO}
      DUP DO DUP DO DUP DO DUP DO DUP DO DUP DO DUP DO DUP DO DO
      }
  Finally, show the copyright notice.  Octal 323 is the code for the copyright symbol.
    1 SETGRAY 0 SETFONT -300 -300 SETXY
    <Copyright \323 1986, Xerox Corporation.  All rights reserved.> SHOW
  }
END
```

Figure 6. The use of geometric transformations to create a spiral effect.

segment, if necessary); this allows the proper joints to be made in place of the end caps.

Several trajectories can be combined to form an outline by using the MAKEOUTLINE operator; an outline represents a filled geometric shape, which may have holes (using multiple trajectories that wrap in opposite directions). An outline can be used with the MASKFILL operator to fill the region with the current color. It can also be used as an argument to the CLIPOUTLINE operator to provide a clipping region for all subsequent masking operations (until the imager variables are restored); and a simpler CLIPRECTANGLE operator is also available.

The MASKPIXEL operator uses a bitmap as a mask, represented in the same form as the MAKESAMPLEDBLACK operator. Optionally, it can be in a compressed format.

Figure 5 shows an example that uses many of the Interpress graphics primitives to draw an ice cream cone. The written encoding of the entire master is shown along with comments. You may want to try plotting the positions of the vertices and control points on a piece of graph paper to get a better feel of how they work.

One of the more useful capabilities of Interpress is the set of geometric transformations that are at the heart of the imaging operations. Interpress has a set of operators for building primitive transformations (SCALE, ROTATE, TRANSLATE) and two operators for combining transformations (CONCAT, CONCATT). Mathematically, the transformations can be represented as 3×3 matrices. Transformations can be applied to any graphic image, including line drawings, scanned images, and character shapes. The transformation capabilities are also useful in publishing applications such as two-sided printing of bound documents and creation of N-up printing signatures.

Figure 6 shows the use of geometric transformation to create a spiral.

Page independence

One key attribute of the Interpress document structure is its page independence. Page independence or page isolation means that the language description of each page is totally independent of that of any other page. With page independence, documents can be decomposed and printed in arbitrary page order. (Many printers find it desirable to print last page first. Two-sided printing may require unusual page printing sequences.) Page independence also enables users to build utility routines for the purpose of creating a new Interpress master from pieces of existing masters, or for creating two-up, head-to-toe, or signature masters used in publishing environments.

Open system architecture

Interpress has evolved from many years of use in a distributed network environment. The Interpress architecture, with its functional richness in describing pages and documents and its device-independent capability, represents an effective solution to the problem of printing a wide variety of documents in diverse printing environments with a common standard interface. Interpress has proven to be gracefully extensible to new printing technologies, new applications, and other new requirements as they emerged, allowing existing masters to be used without modification.

Xerox has chosen to make the Interpress printing architecture a completely open system. Interpress and related systems have been made publicly available, free of any royalty or licensing fees.

Abhay Bhushan is the manager of System Integration Standards for Xerox Corporation, where he has worked since 1974. Prior to joining Xerox he worked as a member of the research staff at MIT's Project MAC on the ARPANET project, and chaired the group developing file transfer and mail protocols.

Bhushan received his BSEE from the Indian Institute of Technology in Kanpur in 1965, an MSEE from the Massachusetts Institute of Technology in 1967, and a master's degree in management from the MIT Sloan School in 1971.

Michael Plass earned his bachelor's and master's degrees from Northern Illinois University, and his PhD in computer science in 1981 from Stanford University. Since 1981 he has worked at the Xerox Palo Alto Research Center. His research interests include computer graphics and composition, as well as algorithms and data structures.

Readers may write to the authors at Xerox Corporation, 3333 Coyote Hill Road, Palo Alto, CA 94304.

COMPUTER SCIENTISTS

The Office of Naval Research is seeking two highly qualified individuals to plan and manage programs of sponsored research in the computer science fields of software engineering and/or architectures and distributed computing. The research sponsored is conducted principally at universities and industrial laboratories. These are Civil Service positions at the GM–13, GM–14 or GM–15 ($37,599–$67,940) level, depending on individual qualifications.

The responsibilities for managing the programs include recognizing and defining new research opportunities in the above areas. Operating with available resources, the manager receives and evaluates proposals and selects for funding those which contribute to the strongest research program. The incumbent maintains close liaison with the ongoing research, monitors the results for information of importance to Navy interests, communicates these results to appropriate Navy activities, and manages follow-on developmental work. An inherent job satisfaction in the position is obtained through the opportunity to have a creative and significant impact on the quality of research conducted at the national level.

Applicants must have either (1) a Ph.D. or equivalent in computer science, mathematics, electrical engineering, or a directly related field and at least one year of professional experience or (2) a minimum of three years of progressively responsible professional experience. Demonstrated research experience is preferred.

Interested persons should submit a resume or Standard Form 171, Personal Qualifications Statement (available at Federal Job Information Centers or from the address below), to:

OFFICE OF THE CHIEF OF NAVAL RESEARCH
Civilian Personnel Division, Code 01242P
ATTN: Announcement #86-40 (IC)
800 North Quincy Street
Arlington, VA 22217-5000

Applications will be accepted through 18 July 1986 and must be received by that date. Applicants are requested to complete the appropriate supplemental forms. For further information and supplemental forms, please call (202) 696-4705.

An Equal Opportunity Employer • U.S. Citizenship Required

UPDATE

CS confers top honors at Compcon Spring 86

W. Wallace McDowell Award winner **William D. Strecker** (left) and **Computer Pioneer Medal** winners **David Wheeler** (center) and **John McCarthy** (right) were honored at ceremonies held last March at Compcon Spring 86.

Tse-yun Feng receives the **Richard E. Merwin Award** for Distinguished Service from Computer Society President **Roy Russo.**

A developer of the artificial intelligence language LISP and the principal designer of the DEC VAX architecture were among those receiving top awards from the IEEE Computer Society last March at Compcon Spring 86 in San Francisco.

John McCarthy and David J. Wheeler received the Computer Pioneer medal, cast in honor of 19th century inventor Charles Babbage and awarded to outstanding individuals whose main contribution to the concepts and development of the computer field was made at least 15 years earlier.

McCarthy, director of the Artificial Intelligence Laboratory at Stanford University, coined the term "artificial intelligence" while working on a research project at Dartmouth College in 1956. He worked on the development of LISP and contributed the "reasoning program" approach to AI in a 1958 paper. This approach involves representing facts about the world and the problem a program is to solve as sentences of mathematical logic. The program then uses directed search procedures to deduce that a certain course of action will achieve the desired goal.

McCarthy also proposed the concept of time-sharing in 1957 while working at MIT.

Wheeler, professor of computer science at Cambridge University, was creator of the first wired-in assembler, built for the EDSAC computer. He invented the concept of "closed subroutines," paving the way for current ideas in modular programming. From 1951 to 1953 he worked at the University of Illinois where he laid the foundations of the programming systems of the ORDVAC and ILLIAC computers. He returned to Cambridge in 1953 and in 1957 designed the programming system for EDSAC-2, a microprogrammed computer.

William D. Strecker of Digital Equipment Corporation received the W. Wallace McDowell Award for outstanding contributions to the computer field. Established through a grant from IBM, the award has been conferred in the past on industry notables such as Seymour Cray, C. Gordon Bell, Gene Amdahl, and Donald Knuth.

Strecker joined DEC in 1972 where he conducted research on cache memories, leading to the development of the PDP-11/70. He served as principal designer of the VAX-11 architecture and also played a major role in the development of the high-performance computer interconnect and the systems communications architecture for VAX clusters.

Tse-yun Feng received the Richard E. Merwin Award, the Society's highest service honor. Named in honor of the 1981 past president of the Computer Society, the award recognizes outstanding service to the Society, to the IEEE and other organizations on behalf of the Society, or to the computer profession. Feng is a past president of Computer Society, the outgoing editor-in-chief of *IEEE Transactions on Computers*, and a former governor of AFIPS.

Computer Society award winners for 1985

Technical Awards

W. Wallace McDowell Award

William D. Strecker, for his contributions as principal designer of the VAX architecture.

Computer Pioneer Award

John G. Kemeny, for the development of BASIC.

John McCarthy, for the development of LISP.

Alan Perlis, for work in computer language translation.

Ivan Sutherland, for the development of Sketchpad.

David Wheeler, for developing the first assembly language system with library subroutine capability.

Heinz Zemanek, for building MAILUEFTERL, the first Austrian electronic computer.

Computer Entrepreneur Award

William Norris, founder and chairman emeritus of Control Data Corporation.

Kenneth H. Olsen, founder and president of Digital Equipment Corporation.

Eckert-Mauchly Award

John Cocke, for contributions to high performance architecture through look-ahead, parallelism, and pipeline utilization, and for contributions to reduced instruction set computer architecture through the exploitation of hardware-software trade-offs and compiler optimization.

Technical Achievement Award

Algirdas Avizienis, for sustained contributions to the area of fault tolerant computing.

Service Awards

Richard E. Merwin Distinguished Service Award

Tse-yun Feng

Distinguished Service Certificate

Rolland B. Arndt, for leadership in membership and information, for systematizing the Computer Society's procedures, and for keeping its by-laws current.

Taylor L. Booth, for leadership in originating the Computing Sciences Accreditation Board.

Oscar N. Garcia, for many years of outstanding leadership in representing the Computer Society.

Edward A. Parrish, Jr., for long-term contributions to technical activities of the IEEE Computer Society, to IEEE transnational relations, and for chairing the first China Conference.

Roy L. Russo, for excellence in leadership of conferences and tutorials; for conception, initiation, and outstanding initial editorship of *IEEE Design & Test of Computers* magazine.

Outstanding Contribution Certificate

Theodore H. Bonn, for representing the Computer Society to the Institute.

Taylor L. Booth, for contributions to computer science and the profession through the generation of criteria for accrediting computer science programs.

Paul L. Borrill, for innovative service to the Board of Governors and to the Computer Society.

Fletcher J. Buckley, for excellence in contributions to IEEE standards.

James T. Cain, for contributions to computer science and the profession through the generation of criteria for accrediting computer science programs.

John Dalphin, for contributions to computer science and the profession through the generation of criteria for accrediting computer science programs.

Dennis W. Fife, for service to Computer Society operations and the development of Compmail +.

George I. Davida, for contributions to computer science and the profession through the generation of criteria for accrediting computer science programs.

Gerald L. Engel, for contributions to computer science and the profession through the generation of criteria for accrediting computer science programs.

Norman E. Gibbs, for contributions to computer science and the profession through the generation of criteria for accrediting computer science programs.

David Kniefel, for contributions to computer science and the profession through the generation of criteria for accrediting computer science programs.

Glen G. Langdon, Jr., for effort and leadership in the IEEE Computer Society Model Program in Computer Science and Engineering.

Doris K. Lidtke, for contributions to computer science and the profession through the generation of criteria for accrediting computer science programs.

Raymond E. Miller, for contributions to computer science and the profession through the generation of criteria for accrediting computer science programs.

Michael C. Mulder, for contributions to computer science and the profession through the generation of criteria for accrediting computer science programs.

Merlin G. Smith, for representing the Computer Society to the Institute.

True Seaborn, for excellence in the development of the Society's new magazines.

James N. Snyder, for service to education and Computer Society publications.

Murali Varanasi, for effort and leadership in the IEEE Computer Society Model Program in Computer Science and Engineering.

Stanley Winkler, for representing the Computer Society to AFIPS and the National Computer Conference Board.

Helen M. Wood, for creative excellence in the Society's fiscal management.

Michael Wozny, for outstanding leadership of *IEEE Computer Graphics & Applications* magazine.

Stephen S. Yau, for leadership of the award-winning *Computer* magazine.

Meritorious Service Certificate

Marshall D. Abrams, for outstanding leadership in computer services to the Society.

Kenneth R. Anderson, for distinguished leadership in Society technical activities.

Kenneth R. Anderson, for many years of continued and dedicated service to the Cherry Hill/International Test Conference and to the Test Technology TC.

James H. Aylor, for innovative leadership in Computer Society Press operations.

Victor R. Basili, for continued service in providing the highest quality of Computer Society technical conferences and their publications.

John E. Bauer, for many years of continued and dedicated service to the Cherry Hill/International Test Conference and to the Test Technology TC.

Daniel Benigni, for service to the Computer Society's Washington Chapter.

P. Bruce Berra, for service to the Board of Governors and to the Computer Society Press.

J. Thomas Cain, for leadership in educational activities.

David E. Carlson, for leadership in setting the IEEE 802.2 standard for local area network logical link control.

Christina Champion, for the Chapter Activities Newsletter and membership promotion.

Robert A. Donnan, for leadership in setting the IEEE 802.5 standard for the token ring local area network.

Robert H. Douglas, for leadership in setting the IEEE 802.4 standard for the token bus local area network.

T. Michael Elliott, for excellence in strategic planning and Computer Society operations.

Joseph Fernandez, for many years of service to West Coast operations.

Herbert Freeman, for many years of distinguished service to the Society.

Sakti P. Ghosh, for many years of dedicated service to the Society's educational activities.

William Habingreither, for extraordinary service to both the Society's West Coast and East Coast offices.

Edith Hayman, for many years of outstanding salesmanship of the Computer Society.

Herbert Hecht, for many years of distinguished service to the Society's publications and its Board of Governors.

Ronald G. Hoelzeman, for leadership in the Society's switch to electronic publishing.

Samuel Horvitz, for many years of continued and dedicated service the Society's technical activities and publications.

Gerrie Katz, for many years of distinguished service to Computer Society conferences.

Frieda Koester, for energetic leadership in the Society's press marketing and West Coast operations.

Ned R. Kornfield, for leadership in the Society's Technical and Professional Activities.

Mercy Kowalczyk, for organization of the Society's Washington office and for service to the Governing Board.

Donald C. Loughry, for leadership in setting the IEEE 802.3 standard for CSMA/CD local area network.

Stephen F. Lundstrom, for creative managerial contributions to the Society's West Coast office.

Robert J. Merkert, for long-term dedicated service to the Cherry Hill/International Test Conference.

John F. Meyer, for service in the Technical Activities Board, the Board of Governors, and for leadership of the Elections Committee.

C. Thomas Mitchell, for many years of dedicated service to the Cherry Hill/International Test Conference.

John D. Musa, for innovative leadership of Computer Society publications.

Raymon P. Oberly, for many years of dedicated service to the Cherry Hill/International Test Conference and to the Test Technology TC.

Hillel Ofek, for exemplary service to the ACM/IEEE Design Automation Conferences and to *IEEE Design and Test of Computers* magazine.

Patricia Paulsen, for outstanding contributions to the Society's awards program and to the organization of its West Coast office.

John T. Polhemus, for many years of dedicated service as local chairman for the Vail Workshop of the Computer Elements Committee.

Marylin Potes, for outstanding leadership in launching *IEEE Design & Test of Computers* magazine and also *IEEE Software*.

Ralph J. Preiss, for distinguished service to the IEEE Computer Society awards program.

Harriett B. Rigas, for distinction in educational activities and for service to the Board of Governors.

John Rini, for service to the Computer Society's Washington Chapter, July 1979 to June 1985.

Peter R. Rony, for creative editorship of *IEEE Micro*.

Susan Rosenbaum, for innovative service to the Computer Society.

William Rosenbluth, for many years of continued and dedicated service to the Phoenix Workshop of the Computer Elements Committee.

Azriel Rosenfeld, for many years of distinguished service to the Society.

Dick B. Simmons, for creative leadership in the Society's planning of joint activities with the ACM.

Robert G. Stewart, for innovations in intersociety cooperation with the ACM and for leadership in the Society's technical and standards activities.

James M. Syck, for establishing the UNOLS SAIL shipboard data communications standard.

Russell B. Theisen, for creative leadership in the Society's area activities and membership services.

Joseph Urban, for creative leadership in the Society's new publications.

Charles R. Vick, for innovations in the expansion of the Society's tutorials and for leadership of its Conferences and Tutorials Committee.

Herbert Weber, for creative leadership in the Society's European activities and for service to the Board of Governors.

Educational Activities Board looks at laboratories

Because the engineering profession emphasizes problem solving and design, students must be trained in a laboratory environment to develop these skills. Rapid advances in computer engineering dictate the need to develop new laboratory courses and periodically update existing facilities and courseware so that this training remains current. This revision effort is normally limited, however, by a lack of resources and manpower, and by problems in conveying to academic administrators the importance of laboratories in a computer science and engineering curricula. Further, the rewards for developing engineering-related laboratories have been limited.

In response to this problem, the Computer Society Educational Activities Board has created a Laboratory Taskforce to work with the academic and industrial communities to make recommendations on needs and directions for laboratory courses. The primary objective is to improve the quality of laboratory education for students in electrical engineering and computer science programs by preparing guidelines for computer-related laboratory courseware along with selected monographs for many of the more standard laboratory sequences. Materials will first be developed for the following areas: introduction to computing, introduction to computer organization, digital systems, intermediate project design, computer architecture, operating systems, computer graphics, software engineering, database systems, AI and expert systems, system simulation, VLSI, concurrent processing, networking and distributed process control.

The laboratory courseware and monographs that result from this project will be published and should be available in December 1986. Those interested in participating in the project should contact one of the committee members listed below.

Keith Barker (Chair)
University of Connecticut (203) 486-2566

A. Wayne Bennett
Clemson University (803) 656-5901

Norm Dillman
Hewlett Packard, Loveland

Mike S. P. Lucas
Kansas State University (913) 532-5600

Dave L. Soldan
Oklahoma State University
(405) 624-6553

Gordon E. Stokes
Brigham Young University
(801) 378-3027

NEW PRODUCTS

Editor: Demetrios Michalopoulos/California State University, Fullerton

Connection Machine has 64,000 processors for 1000 MIPS

Thinking Machines Corp. has announced the Connection Machine system, a massively parallel computer with 64,000 individual processors. According to the company, the system handles information at sustained speeds in excess of one billion instructions per second (1000 MIPS). A series of software applications for the system span a range of word, image, and number problems considered too large or complex for conventional computers.

Thinking Machines asserts that the Connection Machine system reconfigures itself, adapting dynamically to the structure of a problem at the data level. Each element of a data structure is assigned to a separate processor. The processors run in parallel, treating problems in their entirety to achieve data-level parallelism. This is possible because the 64,000 processors, each with its own memory, are dynamically linked through a high-bandwidth communications network. The system forms the connections needed in each application. According to the company, this results in simpler programming because problems need not be broken up into sections for solution. The company predicts that this method of problem solving will revolutionize computer vision, among others.

The **Connection Machine** from **Thinking Machines Corp.** comes in a five-foot cube.

Users interact with the programming environment of a front-end system, such as a VAX or Symbolics 3600 running standard editors, utilities, and operating systems. They write their own programs in extensions of a high level programming language such as C or Lisp, implemented as C* or *Lisp. The compiler and Connection Machine hardware automatically allocate data to the processors, synchronize operations, and determine what data is communicated from one processor to another.

The Connection Machine consists of seven kinds of modules containing special processor chips that implement 16 processors. The system uses 4096 of these chips. Each processor carries out its own fault diagnosis.

The air-cooled system is contained within a five-foot, semi-transparent cube. It comes in two configurations. The machine with 65,636 (64K-byte) processors and 32M bytes of memory costs $3 million. The machine with 16,384 (16K-byte) processors and 8M bytes of memory costs $1 million.

Thinking Machines predicts new markets for Connection Machine applications. In word and language applications, the system allows the use of artificial intelligence algorithms in solving problems. Among the algorithms is a method for searching unstructured databases through whole-document comparison. In picture and vision applications, the system allows image comparison algorithms to operate much faster than in conventional machines, as in computing contour maps from aerial photographs taken from slightly different angles.

Other examples of likely applications include numeric and scientific applications, particularly non-uniform mathematical models where data density varies, or fluid dynamics simulation and linear programming.

Deliveries of the Connection Machine system will begin in July. For more information, contact James Bailey, Thinking Machines Corp., 245 First St., Cambridge, MA 02142-1214; (617) 876-1111.

Reader Service Number 40

FPS T Series uses hypercube-based processors

Floating Point Systems, Inc., has announced the FPS T Series supercomputer. According to the company, the T/10, the smallest T Series configuration, achieves a peak performance of 128 Mflops, while the largest configuration, the T/40000, achieves a peak speed of 262 Gflops. Each model in between reportedly doubles performance from the previous model.

The T Series are parallel processors based on the hypercube topology, with configurations ranging from eight to 16,384 processors. Each processor, or node, reportedly has a peak arithmetic speed of 16 Mflops in full precision (64 bits) with control processing of 7.5 MIPS. Each node has 1M byte of memory and can access up to 1G byte of disk storage. Specifications of the systems result from scaling these features by the number of nodes.

Additionally, each node is controlled by a transputer designed to execute the Occam programming language. FPS adds a library of vector mathematical routines.

According to the company, each module has integral system facilities for fault tolerance, uploading and downloading programs and data, and management tasks independent of application program execution. Effective I/O bandwidth to disk storage is 7M bytes per second for the T/10 and 15G bytes per second for the T/40000.

Prices range from $495,000 at the low end to over $30 million at the high end. For more information, contact Roy Webster, Floating Point Systems, Inc., Box 23489, Portland, OR 97223; (503) 641-3151.

Reader Service Number 41

TI adds business system based on Unix, portable terminals

Texas Instruments Inc. has announced the Business System 1000 Series, based on AT&T's Unix System V. The first member of the series, the Business System 1500, includes multiple 32-bit processors and 4G bytes of memory address space. A Cobol System V programming language provides a compatible Cobol environment between the Business System 1500 and TI's Xenix-based Business-Pro computer.

The **TravelMate Series** from **Texas Instruments** features an LCD, built-in printer, color-coded keyboard, and modem.

The system has a seven-slot chassis and a system bus that permits multiple processors. Each processor board contains a 32-bit 68020 microprocessor. Separate intelligent processors manage mass storage and peripheral input and output. The multiple processors plug directly into a 32-bit NuBus system bus.

Terminal concentrators handle up to 16 terminals each, for a total of 128.

The Business System 1500 supports TI's Model 931 video display terminal, or the Model 924 VDT. Moreover, TI PCs and the TI Business-Pro can be used as terminals via terminal emulation, according to the company.

The operating system, TI System V, is based on AT&T's System V. It supports Cobol, Pascal, C, and 68000 assembly language. The Cobol development environment includes a Cobol compiler, a source-level Cobol debugger, run-time environments, a forms, designer, and a sort/merge package.

The Business System 1500 will begin shipping in the third quarter of 1986. The list price of the twelve configurations available will range from $70,000 to $408,000. Contact Texas Instruments Inc., Data Systems Group, PO Box 809063, H-850, Dallas, TX 75380-9063; (800) 527-3500.

Texas Instruments also announced the addition of three portable display terminals to its Silent 700 Series. The TI Silent 700 TravelMate Series includes the TravelMate ($1095 suggested), TravelMate 1200 ($1295), and TravelMate DT ($995). All three feature a retractable, 16-line by 80-column LCD screen, built-in 45 cps thermal printer, keyboard, data communications interfaces, and application and text editing functions. The first two come with internal 300-baud and 300/1200-baud modems, respectively, while the TravelMate DT, a desktop version, features an EIA RS-232-C interface that supports up to 9600-baud communications. They will be available pending FCC approval. Contact the same address as above, but send inquiries to PO Box 809063, H-849.

Reader Service Number 42

32-bit supermicro based on VMEbus

Dual Systems Corp. has announced Chaparral, a 32-bit supermicrocomputer based on VMEbus architecture and built on the 16-MHz MC68020 microprocessor. The system supports Dual's version of Unix, which combines elements of System V and Berkeley 4.3, and the programming languages Pascal, Fortran-77, Cobol, Basic, and Lisp. It includes AT&T's System V C compiler, program generators, debuggers, and other utilities.

Each VMPU-32 module includes its own MC68020 CPU, an MC68851 coprocessor for paged memory management, an MC68881 floating-point coprocessor, and 1M byte of dual-port RAM. Top-end configurations include Dual's VESDI (VMEbus Enhanced Small Device Interface) board or VSMD (VMEbus Storage Module Device) board.

A basic system including card cage, power supply, system arbiter, and three VMEbus modules (VMPU, VIOP, and VUSC) plus hardware and documentation costs from $14,900 to $26,900. Chaparral is sold through original equipment manufacturers (OEMs), value-added resellers (VARs), systems integrators, and distributors. Dual Systems Corp. is located at 2530 San Pablo Ave., Berkeley, CA 94702.

Reader Service Number 43

Touch technology for tests

Interaction Systems, Inc. has announced a stand-alone capacitive touch technology evaluation kit. The ISI Model 3100 Touch Controller Kit includes a Series 3000 capacitive touch screen, serial controller, and power supply. It plugs into any asynchronous port. The kit features self-checks, diagnostics, and communications loop back tests. It utilizes a standard Model 3002 production controller board. The kit is priced from $595, depending on the screen size (9 to 19 inches). Contact Interaction Systems, Inc., 24 Munroe St., Newtonville, MA 02160; (617) 964-5300.

Reader Service Number 44

Flexible's C2C joins C1C

Flexible Computer Corp. has announced the C2C, a 32-bit real-time computer for the Flex/32 Multicomputer. It was designed for Motorola's 16 and 20 MHz 68020 processor and will function along with Flexible's C1C, which uses National Semiconductor's 32-bit 32032 processor.

The C2C employs a three-ported, asynchronous design. It supports automatic memory remapping and scatter/gather memory mapping. The system's fair access arbitration scheme splits memory bandwidth between the processor, intercomputer buses, and the VMEbus, or assigns the entire bandwidth to one function, according to need.

A typical configuration of the Flex/32 Multicomputer with two C2C real-time computers, 2M bytes of memory, an 80M-byte disk, 67M-byte tape, and software costs about $87,000 in OEM quantities. Contact a local sales office or Flexible Computer Corp., 1801 Royal Lane, Bldg. 8, Dallas, TX 75229; (214) 869-1234.

Reader Service Number 45

IBM delivers portable PC

IBM's Entry Systems Division has announced the IBM PC Convertible, a battery-powered computer that weighs less than 13 pounds. It features a detachable display that can be removed and replaced by optional IBM desktop displays, and an optional modem.

The IBM PC Convertible has an 80C88 microprocessor and 256K bytes of memory, an 80-column by 25-line liquid crystal display, a 78-key keyboard, dual 3½-inch disk drives with 720K bytes each, a battery pack, and an AC adapter. Memory can be expanded to 512K bytes with the addition of 128K-byte memory cards.

The IBM PC Convertible with 256K bytes of memory, dual disk drives, LCD, keyboard, battery pack, adapter, and start-up software costs $1995. Options include a matrix printer that attaches to the back of the Convertible and draws its power from the unit's battery pack. The only version of DOS that supports the Convertible, DOS 3.2, costs $95.

IBM has also announced a new model of the PC-AT, three new models of the PC-XT, software for communication between PCs and host computers, two series of accounting software, and lower prices on a variety of products.

The IBM PC 3270 Emulation Programs, Entry Level ($185) and Version 2 ($425), and the IBM 3270 PC Control Program, Version 3 ($450), allow IBM PCs to communicate with large systems as stand-alone workstations, according to the company. An enhanced version of the PC Network Program, renamed the IBM PC Local Area Network Program Version 1.1 ($125), supports IBM PCs on both the IBM PC Network and the IBM Token-Ring Network. It requires DOS 3.2.

The **IBM PC Convertible** has 256K bytes of memory and dual 3½-inch drives.

For more information, contact local IBM sales offices, or IBM's Entry Systems Division, PO Box 1328, Boca Raton, FL 33432; (305) 982-3474.

Reader Service Number 46

New language parser development tool

A.L.P. Systems has announced PeriPhrase, a rule-based linguistic software development environment. According to the company, PeriPhrase suits tasks requiring linguistic processing, such as the development of natural-language interfaces, computer-assisted instruction, corpus analysis and terminology extraction, and compiler development.

PeriPhrase currently supports development of products for English, French, German, Italian, Dutch, Danish, Swedish, Norwegian, Russian, and Spanish. The company plans support for other languages in the future.

For teaching applications, PeriPhrase generates linguistic parsers to analyze student responses and activate appropriate tutorials when errors are detected.

PeriPhrase can be used to implement compilers for artificial languages, such as expert-system rule languages and other nontraditional programming languages, according to the company. It can be used to generate translators for context-sensitive languages. Attributes can be inherited or synthesized. Annotated syntax or parse trees can be generated and manipulated automatically. Code can be generated by tree rewriting or semantic actions.

A.L.P. Systems has two levels of PeriPhrase for Unix- and Xenix-based environments. Level I consists of the PeriPhrase development environment. Level II consists of the development environment plus one or more source-language modules. A source-language module consists of dictionary support, morphological processing, baseform reduction, automatic sentence segmentation, and character set support.

A.L.P. Systems offers introductory prices through June 15, 1986. For one CPU, compiler and debugger only cost $6000 ($3000 introductory). Level I for multiple CPUs costs $21,000 ($11,000 introductory). Level II for a single CPU costs $12,000 ($6000 introductory); for multiple CPUs, $42,000 ($21,000 introductory).

For more information, contact A.L.P. Systems, 190 West 800 North, Provo, UT 84604; (801) 375-0090.

Reader Service Number 47

MDB Systems chassis and three-board set expand MicroVAX II

MDB Systems, Inc. has announced two expansion products for MicroVAX II computers. The MLSI-BA11-EXP allows the addition of six quad size or 14 dual size modules onto the MicroVAX II computer. It consists of an expansion chassis (MLSI-BA11-3000) and a three-board expansion set (MV-DB11-E) available by itself.

According to the company, the Q-Bus is repeated over a distance of 10 feet, allowing the insertion of modules totaling up to 20 DC bus loads into the expansion chassis. A dual size bus repeater module is inserted into the MicroVAX II chassis with a second dual size bus repeater module inserted into the first slot and a dual size bus grant module inserted into the last slot of the expansion chassis.

The three-board expansion set comes with three 10-foot shielded cables. The expansion chassis with which the boards are to be used must have provision for an unused 'A' size option plate. Sold by itself the set costs $1945. The MLSI-BA11-EXP costs $3595. Discounts are available for OEMs and large-volume end users.

For more information contact MDB Systems, Inc., 1995 N. Batavia St., Orange, CA 92665; (714) 998-6900.

Reader Service Number 48

New-release software for IBM PCs searches text

ZyLab Corp. has announced a new release of its full-text search software for IBM PCs and compatibles. The ZyIndex 2 is compatible with compact disk read-only memory devices (CD-ROMs), write-once-read-many optical disks (WORMs), local area networks (LANs), and 26 word processors, including IBM DisplayWrite 2 and 3, Spellbinder, Officewriter, WordPerfect, Multimate, and WordStar. The software also displays foreign-language or extended ASCII characters.

The Plus version provides simultaneous multiuser read and write access to network file servers on IBM Token Ring, Novell, and 3Com LANs and is compatible with other MS-DOS 3.1- and 3.2-based LANs. Both the Plus and Professional versions support selective searching.

The ZyIndex 2, handling 500 files, costs $145; the Professional version, handling 5000 files, costs $295; and the Plus version, handling 15,000 files, costs $695. Tutorial disks plus booklet cost $10. Current users can upgrade to Release 2 for $35.

For further information, contact ZyLab Corp., 233 E. Erie St., Chicago, IL 60611; (312) 642-2201.

Reader Service Number 49

IMSL Library and SFUN/Library for MicroVAX I and II and IBM RT PC

IMSL has released the IMSL Library and SFUN/Library for MicroVAX I and II computers using the VAX-11 Fortran compiler running under the MicroVMS operating system. The libraries are also available on Digital Equipment's VAX-11 mainframe computers under the VAX/VMS operating system.

IMSL Library offers over 500 mathematical and statistical Fortran subroutines. SFUN/Library offers Fortran subprograms for evaluating elementary and fundamental functions, Bessel functions, exponential integrals, error functions, trigonometric and hyperbolic functions, and gamma functions.

The IMSL Library's annual license fee on the MicroVAX I and II is $1200 for the first year and $1000 for renewal. The SFUN/Library's fee is $850 for the first year and $550 for renewal. Organizational and paid-up licenses are available, as are reduced prices for educational institutions and discounts for quantity orders. Contact IMSL Sales Division, 2500 ParkWest Tower One, 2500 CityWest Blvd., Houston, TX 77042-3020; (713) 782-6060.

IMSL also offers the two libraries plus the Protran problem-solving system for the IBM RT PC using the IBM RT PC Fortran-77 compiler, under the Advanced Interactive Executive (AIX) operating system. Protran handles problem-solving in mathematics, statistics, and linear programming, according to the company. License prices range between $850 and $1200 for the first year, renewable between $550 and $1000.

Reader Service Number 50

GUPI helps program programmable devices

Intel Corp.'s iUP GUPI (Generic Universal Programmer Interface) lets designers program programmable devices, including erasable programmable read-only memories (EPROMs), erasable programmable logic devices (EPLDs), and programmable microcontrollers. It supports a variety of package types, including pin grid array, plastic leaded chip carriers, and up to 68-pin packages. The product works with Intel's existing line of universal PROM programmers, or plugs directly into Intel's iPDS Personal Development System. It also supports Intel's CHMOS EPLDs and EPROMs, plus Intel's 87C51, 8795, 8796, and 8797 microcontrollers. The iUP GUPI costs $790, while adapters cost $200 to $250 each.

For more information, contact Intel Corp., Literature Dept., W-295, 3065 Bowers Ave., Santa Clara, CA 95051.

Intel's iState state machine software according to the company allows systems hardware designers using EPLDs to develop a state machine design and enter the design into the Intel Programmable Logic Development System (iPLDS). The software supports multiple syntaxes for state definition, specification of state transitions, inputs, and outputs. It also provides intermixing of state machine and Boolean equation designs and the use of gates such as D, JK, SR, or T flip-flops in designs. Upgrades are available for $500. The complete iPLDS sells for $2500. Designers who already have an iUP series programmer can purchase the iPLS software for $1450. For additional information, contact Intel's Literature Dept., W-298.

Intel also announced the addition of four EPLDs and supporting CAE software: the 1800-gate 5C180 and 900-gate 5C090 EPLDs; two versions of a 300-gate EPLD, the 5C031 and 5C032; a revision of the iPLDS software; and programming hardware compatible with the EPLD packaging. For more information, contact Intel's Literature Dept., W-296.

Reader Service Number 51

LMI ObjectLisp in public domain

Lisp Machine, Inc. (LMI) has announced ObjectLisp, a portable, second-generation, object-oriented programming paradigm written in Common Lisp. Features include portability, software development in the ZetaLisp environment, implementation in Common Lisp, and public domain status. The paradigm uses the same syntax as normal Lisp and employs function calls.

ObjectLisp is available on Lambda, the LMI AI workstation, and runs on non-Lisp computer systems such as Sun 2 and Sun 3 workstations running Sun Common Lisp (Lucid); Silicon Graphics Iris Workstation (Franz); IBM PC-AT (Gold Hill); DEC VAX/VMS; DEC MicroVAX running DEC Common Lisp; Symbolics 3600; and the Texas Instruments Explorer.

The software comes on a half-inch magnetic tape packaged with a user's guide and source code. A shipping, handling, and media charge of $195 applies to the first copy. Subsequent copies may be made without any obligation to LMI.

For further information, contact Marketing Dept., Lisp Machine, Inc., 6 Technology Dr., Bldg. #4, Andover, MA 01810; (617) 682-0500.

Reader Service Number 52

Robot controller under development

Univision III from **Unimation Inc.** identifies stationary objects.

Unimation Inc. is developing the Unival robot control system, also called Val III. According to the company, the system features up to eight-millisecond processing speed, all-digital servo and torque control, a modular configuration, the Val III programming language, and two plug-in vision system options, Univision III and Conveyorvision. Unival is reportedly compatible with standard local-area networks such as MAP, SECS III, and Intel's Bit Bus.

Unival is available in two standard and expandable configurations: the Basic configuration for single-robot operations, and the Standard configuration for multi-robot/manipulator control in single workcells.

Unival's capabilities include two-dimensional array data manipulation; use of integers, real values, variables, transformations, and expressions; file management; programming constructs; battery-backed CMOS memory; continuous path and point-to-point programming in joint, world, tool, and workpiece modes; and real-time path modifications.

Univision III is an area vision system to identify and locate stationary objects. Conveyorvision is a line-scanning vision system to identify moving objects on a conveyor and determine their locations relative to the conveyor. Both can identify touching and overlapping parts and come with preprogrammed software.

The company anticipates releasing Unival this year. For more information, contact Unimation Inc., Shelter Rock Lane, Danbury, CT 06810; (203) 796-1069.

Reader Service Number 53

Puma CAD package targets Apple Macintosh

Puma Software, Inc. has announced Timdom/Mac, a computer-aided design package for control system design and analysis for the Apple Macintosh personal computer. The program consists of menus of programs for engineering mathematical functions, system analysis, design, estimation and filtering, and model reductions. It can produce two-dimensional graphics of relationships. The program is written in MS-Basic.

Mathematics functions include routines to compute the Eigenvalues and Eigenvectors of matrices; calculate matrix inverse, rank, and norm; find roots of complex polynomials; perform Runge-Kutta integration of differential equations; and perform curve fits of data to polynomials. Timdom/Mac solves Ricatti equations via three methods, calculates inverse Z and LaPlace transforms, and calculates linear equation solution.

Analysis functions include routines to convert continuous systems to discrete form, construct controllability and observability matrices for multivariable linear time-invariant systems, determine stability, calculate transfer functions of multiple input/multiple output systems, determine state equation solutions for continuous and discrete time systems, and calculate companion forms of linear time invariant systems.

Design functions include routines to determine state feedback control laws via pole placement and three types of state regulator problem solvers.

Estimation functions include routines to determine full and reduced order observer state equations, time scale separation, and model aggregation by four methods.

Timdom/Mac runs on a 512K-byte Macintosh PC with an external or hard disk drive recommended. It costs $999.

For more information, contact Puma Software, Inc., PO Box 34373, Albuquerque, NM 87176, or Peggy Andrews, (505) 265-5270.

Reader Service Number 54

Graphics controller for minis and mainframes

Metheus Corp. has announced a graphics display controller for mini and mainframe computers called the Omega 3610, the first in the modular Omega 3000 Series. The controller features 1280-by-1024 resolution with eight to 32 bit-planes; three twelve-in, eight-out lookup tables (LUTs); a custom bit-slice processor; independent text overlay; hardware pan and zoom circuitry; and modular design. According to the company, the structure effectively offers up to 16 eight-in, eight-out LUTs simultaneously (depending on the number of installed bit-planes), with each capable of defining a maximum of 16 million colors.

The base price of the Omega 3610 is $16,950. Contact Metheus Corp., 5510 N.E. Elam Young Parkway, Hillsboro, OR 97124; (503) 640-8000.

Reader Service Number 55

Recent Microsystem Announcements

For more information, circle the appropriate RS No. on the Reader Service Card at the back of the magazine.

MANUFACTURER AND MODEL	FUNCTION	COMMENTS	R.S. NO.
Analogic Corp., ITG 1105, ITG 1121, ITG 1130, ITG 1110	I/O boards	Multibus I-compatible I/O boards with A/D conversion modules and varied numbers of input and output channels. ITG 1105, ITG 1121, and ITG 1130 replace Intel's SBC 711, SBC 724, and SBC 732 boards, respectively. ITG 1110 upgrades Intel's SBC 711. Cost: $680 for ITG 1105, $640 for ITG 1121, $845 for ITG 1130, and $1395 for ITG 1110. Analogic Corp., Industrial Technology Group, 8 Centennial Dr., Centennial Industrial Park, Peabody, MA 01961; (617) 246-0300.	80
CMS, Rocky I	Hard disk controller	A hard disk drive controller for IBM PCs and compatibles running MS-DOS Version 2.0 or higher or IBM PC-DOS. Uses the ST-506 disk drive interface and supports one or two drives, each with up to 16 heads and 1024 cylinders. Includes an on-board ROM BIOS. Cost: $95 (1000s). CMS, 401-B W. Dyer Rd., Santa Ana, CA 92707; (714) 549-9111.	81
Communication Machinery Corp., Internet, ENP 50, ENP 44	Protocol software, Ethernet node processors	Software and hardware to connect VAX and MicroVAX systems through Ethernet. ICMP, UDP, ARP, TCP, and IP protocol modules reside on the Ethernet node processor (ENP) boards. ENP 50 conforms to Ethernet 1.0 and IEEE 802.3 specifications and links MicroVAX computers to the LAN. ENP 44 links VAX computers to the LAN. Cost: $2500 for Internet, $2500 for ENP 50, and $3500 for ENP 44. Communication Machinery Corp., 1421 State St., Santa Barbara, CA 93101; (805) 963-9471.	82
Electronic Vaults, Inc., UPTA 96 series	Modems	9600-baud intelligent asynchronous modems that operate over dial-up telephone lines. UPTA 96 Integral is an integral piggy-back card for a short or full expansion slot on IBM PCs or compatibles. UPTA 96/S, a stand-alone model, has an RS-232 connector for PC and micro to mainframe applications. Cost: $895 for UPTA 96 Integral, $995 for UPTA 96/S. Electronic Vaults, Inc., 12347-E Sunrise Valley Dr., Reston, VA 22091; (703) 620-3900.	83
Excelan Inc., EXOS 8052-01 NETBIOS-TCP/IP, EXOS 8051-02 TCP/IP	Software for Ethernet	Run on TCP/IP protocol and on Excelan's EXOS 205 Intelligent Ethernet Controller Board, and use DOS. EXOS 8052-01 NETBIOS-TCP/IP provides compatibility with IBM's NETBIOS over an Ethernet network. EXOS 8051-02 TCP/IP includes a socket library for PC-to-host networking applications. Cost: $95 for EXOS 8052-01 NETBIOS-TCP/IP, $595 for EXOS 8051-02 TCP/IP with socket library, FTP, and Telnet. Excelan Inc., 2180 Fortune Dr., San Jose, CA 95131; (408) 434-2300.	84
Ideassociates, Inc., Idea Supermax/EMS	Multifunction board	A multifunction board for the IBM PC-AT with expanded, extended, and conventional memory; two serial ports; one parallel port; and memory up to 4M bytes. Includes print spooling software. Cost: ranges from $495 for a bare board to $2595 for 4M bytes of memory. Ideassociates, Inc., 35 Dunham Rd., Billerica, MA 01821; (617) 663-6878.	85
Imaging Technology Inc., Series 151 Image Processor	Image processor	A VMEbus-based subsystem that connects to the IBM PC-AT. Includes a hardware implementation of area-of-interest processing. Uses four boards: FB-150 Frame Buffer, ADI-150 A/D Interface, ALU-150 Pipeline Image Processor, and RTC-150 Real-Time Convolver (optional). Comes with Toolbox 151, a software applications development package. Cost: $11,495. Imaging Technology Inc., 600 West Cummings Park, Woburn, MA 01801; (617) 938-8444.	86
Lifeboat Assoc., Run/C Professional	C interpreter	Follows Run/C, the C Interpreter. Adds loadable libraries, source-level debugging facilities, language implementation of Kernighan and Ritchie C (lacks preprocessor conditionals and undefine), and built-in editor. Available for IBM PC, PC-XT, PC-AT, and compatibles, plus other systems with MS-DOS and ANSI terminal drivers. Cost: $250; $85 as upgrade for users of Run/C through version 1.32. Lifeboat Assoc., 1651 Third Ave., New York, NY 10128; (212) 860-0300.	87
Light Machines Corp., Linedrive 3000	Control system	Permits the IBM PC, PC-XT, and compatibles to operate a three-axis DC servo positioning system in industry standard RS-274D NC code. Off-line storable programs. Consists of software and interface card with optional DC servo amplifiers and motors. Cost: $4000. Light Machines Corp., 669 E. Industrial Dr., Manchester, NH 03103; (603) 625-8600.	88

MANUFACTURER AND MODEL	FUNCTION	COMMENTS	R.S. NO.
Matrox Electronic Systems Ltd., VG-640	Graphics controller	A single dual-height VMEbus card that drives a 60 Hz noninterlaced display. Combines the Hitachi ARCTC graphics controller with a 2D/3D command set executed by an on-board 32/16 bit CPU. Has 8 bits/pixel and can display 256 colors from a palette of 262,144. Cost: $2995. Matrox Electronic Systems Ltd., 1055 St. Regis Blvd., Dorval, Quebec H9P 2T4, Canada; (514) 685-2630.	89
Oregon Micro Systems, Inc., PC68, RS68	Motion control subsystems	68008-based multi-axis motion control subsystems. PC68 interfaces any IBM PC with up to 10 stepping, linear, or servo motor drives that accept digital step and direction commands. RS68 interfaces any RS-232 or RS-422-based communication link with up to 6 stepping, linear, or servo motor drives. Cost: $1495 for 2-axis PC68; $1295 for 2-axis RS68. Oregon Micro Systems, Inc., 15075 N.W. Pioneer Rd., Beaverton, OR 97006; (503) 644-4999.	90
Plessey Microsystems, PME SCSI-1	SCSI board	An intelligent VMEbus small computer systems interface (SCSI) board with an on-board 68008 processor. Capable of asynchronous SCSI bus transfer rates of up to 1.5M-bytes/s with DMA transfers of up to 32 bits over the VMEbus. Compatible with both VMEbus and ANSI X3T9.2 SCSI specifications. Cost: $1660. Plessey Microsystems, One Blue Hill Plaza, Pearl River, NY 10965-8541; (914) 735-4661.	91
Quadram, QuadEMS +	Expanded memory board	An IBM PC-XT expanded memory board. Software switchable between AQA EEMS (AST/Quadram/Ashton-Tate Enhanced Expanded Memory Specification) and LIM EMS (Lotus/Intel/Microsoft Expanded Memory Specification). Supports 64K- and 256K-byte chips on the same board at the same time. Comes with 256K bytes of memory. Cost: $495. $545 for QuadEMS + I/O, a multifunction version. Quadram, One Quad Way, Norcross, GA 30093-2919; (404) 923-6666.	92
Rapid Systems, Inc. R412 Data Logger Peripheral	Data logger peripheral	A data logger peripheral for IBM PC, PC-XT, PC-AT, and compatible computers. Hardware features: 8-bit A/D converter, external/internal/event driven trigger, BNC input connections, gain ranges from 0.8V to 160V peak in eight steps, and diode-protected inputs. Software features: automatic save, strip chart printing, auto scrolling, and digital oscilloscope software. Cost: $688. Rapid Systems Inc., 755 N. Northlake Way, Seattle, WA 98103; (206) 547-8311.	93
Spencer Organization, Inc., APL Interpreters	ALP interpreters	Four APL interpreters based on APL.68000. Come in single or multiuser versions, 10 or 12 MHz, 512K bytes or 1M byte. Prices start at $995. Spencer Organization, Inc., Box 248, 366 Kinderkamack Rd., Westwood, NJ 07675; (201) 666-6011.	94
SPS Software Products & Services, Inc., EPOS	Project design and development software	Software for project design, development, and reporting on the IBM PC, PC-AT, and compatibles under MS-DOS 3.0 and up. Supports seven design methods, switchable during the project. Automatically generates text and graphic documentation. Includes integrated management functions. Cost: about $14,200 per PC. SPS Software Products & Services, Inc., 14 E. 38th St., 14th Floor, New York, NY 10016; (212) 686-3790.	95
Standard Microsystems Corp., SMC4030-PC	Controller board	Controls up to two standard Winchester disk drives, two IBM-compatible floppy disk drives, and one QIC-02 streaming tape drive. Plug compatible with the IBM PC-XT bus. Features split mode compatibility, data error detection and correction, and on-board data separation. Cost: $275 (1-99). Standard Microsystems Corp., 35 Marcus Blvd., Hauppauge, NY 11788; (516) 273-3100.	96
Star Gate Technologies, MRB-2000	Expanded memory board	For the IBM PC-AT and compatibles operating under multiuser, multitasking operating systems. Uses standard 256K-byte, 150-ns RAM memory chips. Cost: $282 for 0K bytes, $429 for 512K bytes, $577 for 1M byte, $725 for 1.5M bytes, $872 for 2M bytes. Star Gate Technologies, Inc., Suite 109, 33800 Curtis Blvd., Eastlake, OH 44094; (216) 951-5922.	97
Techware, Pangloss	Software	Software for multilingual word processing with WordStar. Four alphabets available: Arabic, Greek, Hebrew, and Russian. Normal English ASCII plus 94 special characters. For 8-bit systems. Cost: ranges from $169 to $199 depending on the system. Techware, 474 Willamette St., Suite 201, PO Box 10545, Eugene, OR 97440; (503) 484-0520.	98
Wendin, PCUnix	Operating system	A version of AT&T's Unix for the IBM PC and compatibles. Over 70 commands. Incorporates Wendin's version of the Bourne shell and includes complete source code. Comes on four disks and runs on a 5M-byte hard disk. Runs MS-DOS and PC-DOS programs. Cost: $99. Wendin, Box 266, Cheney, WA 99004; (509) 235-8088.	99

Recent IC Announcements

For more information, circle the appropriate RS No. on the Reader Service Card at the back of the magazine.

MANUFACTURER AND MODEL	FUNCTION	COMMENTS	R.S. NO.
Honeywell, HE8000	Bipolar ASIC	An 8000-gate array, 1.25-micron, bipolar, application-specific IC that features current mode logic, on-chip testability, programmable-drive macrocells, and CAD support. Will be available in a 235-pin pin-grid-array package in the fourth quarter of 1986. Cost: $800 (100s). Honeywell Bipolar Product Marketing, Digital Product Center, 1150 E. Cheyenne Mountain Blvd., Colorado Springs, CO 80906; (800) 328-5111.	70
Intech, Inc., IT574A	A/D converter	A 12-bit A/D converter featuring a typical conversion time of 20 μs, 8- or 16-bit microprocessor interface, +10V reference, clock, and three-state outputs. A successive approximation device with four selectable input ranges, available in three grades: 0 to 70 °C (IT574AJD), −55 to +125 °C (IT574ASD), and with Mil-Std-883 screening (IT574ASD/B). Cost: in lots of 100, $34.50 for IT574AJD; $90 for IT574ASD; and $110 for IT574ASD/B. Intech, Inc., Microcircuits Div., 2270 Martin Ave., Santa Clara, CA 95050-2781; (408) 988-4930.	71
National Semiconductor Corp., DM10422A-7	ECL RAM	A 1024-bit emitter-coupled logic (ECL) RAM. Configured as 256 × 4; also configurable as a 512 × 2 or 1024 × 1 device. Voltage compensated and compatible with 10K and 10KH ECL logic. Available in a 24-pin Cerdip and Quad Cerpack. Cost: in 100s, $13 for the Cerdip and $16 for the Cerpack. National Semiconductor Corp., 2900 Semiconductor Dr., PO Box 58090, Santa Clara, CA 95052-8090; (408) 721-5421.	72
Oki Semiconductor, MSM70H000 Series	CMOS gate arrays	Ten 2-micron CMOS gate arrays ranging from 700 to 10,008 gates, with number of I/O's ranging from 66 to 172. Each I/O buffer configurable as input or output, CMOS and TTL compatible. Comes with cell library in dual in-line plastic leaded chip carrier, pin grid array, and flatpack packages. Engineering charges: $13,000 to $50,000. Production pricing ranges from 0.15 to 0.2 cents per gate. Oki Semiconductor, 650 North Mary Ave., Sunnyvale, CA 94086; (408) 720-1900.	73
Precision Monolithics Inc., DAC-8212	D/A converter	A dual multiplying CMOS digital-to-analog converter with a 12-bit-wide data port, internal latches, and a reference-resistance match between DAC A and DAC B of maximum 1 percent. Available in a 24-pin hermetic DIP in two grades (according to accuracy and gain error) over the commercial, industrial, and military temperature ranges. Cost: in 100s, $18.45 and $15.26 for commercial grade; $21.15 and $18.22 for industrial grade; and $72 and $63 for the military grade. Precision Monolithics Inc., 1500 Space Park Dr., PO Box 58020, Santa Clara, CA 95052-8020; (408) 727-9222.	74
Signetics Corp., Military 68000	16-bit microprocessor	A military-qualified 68000 16-bit microprocessor. Available in 64-pin DIP in 6 and 8MHz versions as 68000-6/BXX (DESC Drawing #8202101YA) and 68000-8/DXX (DESC Drawing #8202102YA), respectively. Cost: $346 for 68000-6/BXX and $385 for 68000-8/DXX (100-999). Signetics Corp., 811 E. Arques Ave., PO Box 3409, Sunnyvale, CA 94088-3409; (408) 991-2000.	75
Silicon Systems, SSI 520	Read/write ASIC	A four-channel read/write device for thin-film recording heads. For use with Winchester disk drives and compatible with non-center-tapped thin-film heads. Comes in 24-pin flatpack. Cost: $20 in OEM quantities. Silicon Systems, 14351 Myford Rd., Tustin, CA 92680; (714) 731-7110.	76
Standard Microsystems Corp., COM9026P	LAN controller	A 40-pin, low power plastic version of the COM0926 Arcnet local area network controller. Cost: $32.80 (100s). Standard Microsystems Corp., 35 Marcus Blvd., Hauppauge, NY 11788; (516) 273-3100.	77
Toshiba America, Inc., TMPZ84COOP-6	Z80 microprocessor	A Z80 microprocessor built with CMOS circuitry and a 6-MHz clock. Pin-compatible with DIP-packaged NMOS Z80 devices. Typical operating current of 15 mA. Cost: $3.30 (100s). Toshiba America, Inc., 2692 Dow Ave., Tustin, CA 92680; (714) 832-6300.	78
Weitek Corp., WTL 2516C, WTL 2010B	16-bit integer multipliers	Fabricated on a 1.2-micron double-metal CMOS process. WTL 2516C is a 16-by-16 bit integer multiplier that operates at 38 ns; WTL 2010B is a multiplier accumulator that operates at 45 ns. Cost: $93 for WTL 2516C, $149 for WTL 2010B (100s, dual in-line packages). Weitek Corp., 1060 E. Arques, Sunnyvale, CA 94086; (408) 738-8400.	79

NEW LITERATURE

Reports on expert systems, parallel processing. SEAI Technical Publications puts out *Expert Systems 1986: An Assessment of Technology and Applications* (ISBN 0-89671-074-2) and *Parallel Processing: The Technology of Fifth Generation Computers* (ISBN 0-89671-067-X). *Expert Systems* by Terri C. Walker and Richard K. Miller describes over 475 expert systems classified into 24 categories within manufacturing, business, and professional domains. For each category the authors provide a technology assessment, software inventory, summary of current research, and future outlook. They also review over 40 shells for building expert systems and describe the fundamental technology of expert systems. Appendixes provide addresses and telephone numbers of organizations in the field, plus a glossary. The 369-page book costs $325. *Parallel Processing* explains the workings of several SIMD, MIMD, and dataflow architectures. It also explores the impact of parallel processing and presents projects and products of 35 international research groups and 19 corporations. A survey explores opinions and forecasts. International markets for parallel processing computers are examined for 1986, 1988, and 1990. The report costs $525. Contact SEAI Technical Publications, PO Box 590, Madison, GA 30650; (404) 342-9638.

Computer-aided publishing. InfoVision publishes a 140-page guide to who is selling and what they are selling in the computer-aided publishing market. The *1986 Buyer's Guide to Computer Aided Publishing* details specific products, with references for summaries of important features. Articles discuss the past, present, and possible future of the market. Order for $85 from InfoVision, Inc., 52 Dragon Court, Woburn, MA 01801; (617) 935-6564.

CAD/CAM/CAE. A three-part survey from Daratech/North Holland details the features, capabilities, and limitations of these systems. The three manuals are *CAD/CAM, CAE: The Contemporary Technology, Evaluating Today's Systems, and Survey, Review and Buyers' Guide.* The first volume (ISBN 0-444-86861-5, 255 pages, Dfl. 475) covers the technology. The second volume (ISBN 0-444-86862-3, 120 pages, Dfl. 350) offers a blueprint for evaluating and planning a turnkey CAD/CAM/CAE purchase, including pitfalls. The third volume (ISBN 0-444-86863-1, 700 pages, Dfl. 750) compiles system and industry data, including company profiles, product data, and commentary. Order from Elsevier Science Publishers BV, Book Order Dept., PO Box 211, 1000 AE Amsterdam, The Netherlands. From the USA or Canada, order from Daratech Inc., PO Box 410, Cambridge, MA 02238.

A Corporate Strategy for CAD/CAM/CAE from CyberResearch, Inc., describes how to integrate engineering CAD systems into corporate information systems for productivity improvements. It provides details on choosing the appropriate PC hardware and software for workstations in all departments. The report is free of charge from CyberResearch, Inc., 5 Science Park Center, PO Box 9565, New Haven, CT 06536; (203) 786-5151.

New from Addison-Wesley. *Software Configuration Management: Coordination for Team Productivity* by Wayne A. Babich of Wang Laboratories presents the philosophy of effective software configuration management, a technique for coordinating programmers working together to develop a single product. The goal is to maximize productivity by minimizing wasted effort, duplication, and confusion. The author discusses the use of Unix and Ada tools for software configuration management and provides a list of recommended reading. The 162-page book costs $18.95 (price subject to change).

The Mystical Machine: Issues and Ideas in Computing by John Savage, Susan Magidson, and Alex Stein discusses computer literacy and raises important questions about when and where computer usage is appropriate. The authors based the book on their common experience in revising Brown University's introductory computer course for humanities majors. Cost: $21.95 hardbound. Contact Addison-Wesley Publishing Co., Reading, MA 01867; (617) 944-3700.

Prolog Sourcebook by David K. Lovison tells how to use Prolog to format input and output on terminal displays and printers. It includes source code and defines individual predicates to perform various tasks. The book includes a demonstration program and targets beginners. ISBN 0-934259-10-0, $19.95 paperbound, from Westcomp, Software Engineering Group, 517 N. Mountain Ave., Upland, CA 91786-5016.

VAX software. Digital Equipment Corp. offers the third edition of the *VAX Software Sourcebook*, covering programs for VAX minicomputers and microcomputers. The two-volume set includes nearly 2600 application and systems software programs. Obtain a free copy, order number EJ-28572-46, from Digital Equipment Corp., Printing and Circulation Services, Literature Inquiry Fulfillment, 10 Forbes Rd., Northboro, MA 01532-2597.

New from Howard W. Sams. Howard W. Sams & Co., a division of Macmillan, Inc., has brought out the fourth printing of the *Computer Dictionary* by Charles J. Sippl for $24.95. The book includes a dictionary of basic computer terms and a handbook of computer-related topics.

Advanced Unix: A Programmer's Guide by Stephen Prata discusses key components of the Unix operating system and emphasizes problem-solving. It covers programming features such as the Bourne shell, shell scripts, loops, and system calls. It shows how to create Unix graphics, allocate and structure data in memory, and how to use C in Unix, among other things. The 496-page book retails for $21.95. Order from Howard W. Sams & Co., Dept. R17, 4300 W. 62nd St., Indianapolis, IN 46268; (317) 298-5400.

Developments in U.S. science and technology. The National Science Board publishes *Science Indicators: The 1985 Report* with information about American science, engineering, and technology. Topics covered include support for research and development, industrial science and technology, public attitudes toward science and technology, and advances in science and engineering. It also has chapters on science and mathematics education below the college level and on the role of instrumentation in advancing scientific knowledge. The 330-page report, stock number 038-000-00563-4, costs $15. Send payment to Dept. 36-RY, Superintendent of Documents, Washington, DC 20402; (202) 783-3238 for credit card orders.

"Any clod can have the facts, but having opinions is an art."
Charles McCabe,
San Francisco Chronicle

THE OPEN CHANNEL

RISC for autos

A new automobile design is being hailed by its developers as the industry's first significant architectural innovation since 1893. Called reduced interior seating capacity (RISC), the design is said to make automobiles run faster and more efficiently by reducing the amount of metal in the structure and operating with fewer passengers. Fewer capabilities and less complex design not only result in easier operation, its developers say, but free needed steel-production facilities for other tasks. According to one Bay Area developer, RISC designs contain between 80 and 90 percent fewer wheels than conventional automobiles.

Until the arrival of the RISC design, the trend in standard automobile construction has been and continues to be toward the development of complex motor and fuel controls built into the engine. A typical RISC automobile, on the other hand, limits the power train to a simple set of pedals.

RISC proponents also claim that their machines are especially useful in mass-transit applications. Instead of large buses, which require that all passengers follow the same route, RISC machines allow each rider to be delivered at high speed directly to his destination.

Research is continuing on this novel structure, attempting to solve some of the peripheral difficulties, such as how to steer with only one wheel. Most important, however, will be acceptance not only by the consumer, but by the industry itself. Developers realize that the RISC design will not become standard until a change is deemed necessary. "It will happen as soon as the automobile industry decides to get out of its traditional mode of operation," said one.

R. G. Nelson
NCRI, Inc.

Some euphemisms

Synergy	You do my work.
The learning curve	Take it easy; we'll get there someday.
Advanced technology	It's too complicated for me.
Top-down approach	I'll do the shallow part, you do the difficult.
Modular approach	We buy the subassemblies; the customer pays.
Experience	Obsolescence.
User-friendly documentation	Hail fellow well met approach.
Flexibility	Here a jumper, there a jumper, everywhere a jumper jumper.
RISC fan	I can't remember all the instructions.
Portable design	Don't use the bells and whistles.

Chandan Sen
Research Triangle Institute

Aphorisms for Abacus

In the late 1950s Richard W. Hamming originated the famous aphorism, "The purpose of computing is insight, not numbers." George E. Forsythe later suggested the modification, "The purpose of computing numbers is not yet in sight."

Abacus is sponsoring a competition, soliciting old and new computing aphorisms. Competitors are invited to suggest aphorisms about computing, in matched pairs or singly. If not original, give the author and the documentary source. One entry each, please, of no more than 200 words.

Competition rules: Typewritten postcards, please. One entry only per person, sent to Competition Number 1, *Abacus*, Box 232, Springfield, PA 19604. Entries must be received by July 1, 1986; foreign postmarks will be given one month's grace. The editors' decisions are final, and all entries become the property of *Abacus*. First-prize winners receive two-year subscriptions to *Abacus*; runners up will receive one-year subscriptions. Results and winners' names will appear in the fall issue of *Abacus*.

Book review reviewed

Paul Gray's review of *Thinking about [TLC] Logo* (*Computer*, March 1986) is an example of how one's preconceptions can prevent one from appreciating a book. Gray missed the point that the book is about thinking, logic, and computation, and uses a version of Logo to express ideas.

How shallow can a book be which includes the essential ideas for proving Godel's incompleteness theorem? I have used the book for a freshman honor's seminar in Logo, and my experience is that students are excited and intellectually stretched by the book. Most of the ideas on an object-oriented style of programming can be implemented in any version of Logo. We have used Dr. Logo and IBM Logo, since TLC Logo is not available for the IBM PC.

For Gray to dismiss the book as not worthy of one's time is a disservice to your readers.

Roger B. Kirchner
Carleton College

The Open Channel is exactly what the name implies: a forum for the free exchange of technical ideas. Try to hold your contribution to one page maximum in the final magazine format (about 1000 words).

We'll accept anything (short of libel or obscenity) so long as it's submitted by a member of the Computer Society. If it's really bizarre we may require you to get another member to cosponsor your item.

Send everything to **Jim Haynes, Computer Center, UC Santa Cruz, CA 95065. Bitnet: haynes@ucscc; Arpa: ucscc!haynes@ucbvax.berkeley.edu.**

Confessions of a used program salesman—an update

Since you last heard from me (see *Computer,* April 1982), something has happened to the used program business. While I was busy basking in that warm glow of success that stemmed from finding a way to not reinvent the wheel each time I had to get a product out the door, it seems that a new lady came into town with an even better bag of tricks.

At first I couldn't quite figure out the exact orientation of the object she was peddling. The banter and hoopla that surrounded this universal elixir of hers was confounded by the reported complexity and sheer immensity of the instructions that came with it. Furthermore, the manufacturer's track record was suspect, although the formula they were using consisted of elements of known therapeutic value. Were they in the right proportions? How easy were they to use? How did they all fit together? I decided to investigate.

I scheduled a rendezvous with her and, I must admit, was a bit taken aback by her massiveness and frills, but "You can't judge a book by its cover," I always say, and, besides, she seemed to have all the right stuff in the places that really mattered. I was most interested in her bag of tricks which, she said, contained objects that I would find of value in perpetuating my trade of selling used programs. What she pulled out were *packages.* Not just ordinary packages, like the kind you get on your birthday where you have to guess what is inside. These packages had an envelope on the top of each one which specified what was inside them. Like a flash, I could see the writing on the wall, but in reality it was her writing on those envelopes that lit the fire. I could use the packages as building blocks without looking inside them by just referencing the envelopes for instructions on how to use what was inside. No more looking under the hood of every used program to see what was inside, and how to use it. I could just keep a file of each of these so-called package specifications around to reference.

I asked her what other kinds of talents she had. She told me she was good at juggling, too. Her bag of tricks had some special packages just tailored for doing two things at once. I was impressed at the thought, since there were many of my used programs I could speed up given a mechanism and facility for concurrency. I could run on two 4-cylinder engines instead of a V8 and keep my customers happy.

After this we were making small talk when I asked her if she had any other classy objects to show. She said there was one other package she hadn't shown. I asked her what type it was. She said it was a do-it-yourself type. That I could make it into any type I wanted and it would perform the same function. At first, I was skeptical. How could there be such a generalized routine? But she showed me several instances and I was convinced.

So where does that leave our young hero? What is a gentleman to do? This casual relationship with the lady has turned into a full-fledged courtship. I have fully embraced her way of packaging, and she has taught me to juggle. I am working on a new set of packages to help me in the used program business, and I will keep you posted on how they turn out. I have learned that while you can't always judge a book by its cover, you can pretty well always judge a package by its specification. Ada is, at first look, a complex lady whose reputation has been tarnished by accusations that she is part of the problem, rather than the solution to the software crisis (see "The Emperor's Old Clothes," *CACM*, Vol. 24, No. 2, Feb. 1981). From personal experience, I can honestly say that while she is still a little rough around the edges, she is a welcome partner in the quest for reusable software (the so-called used programs that I purport to sell).

Will Tracz, your friendly used program salesman
Stanford University

News of the Strictly Decimal Initiative

Year after year, the U.S. will lose millions of dollars to useless arithmetic, converting yards into miles and vice versa. The problem is serious enough to threaten the leading role of the U.S. economy. In the computer business, the same mistake is being repeated by defining 1 K of memory as 1024 rather than precisely 1000 words. As usual, the eggheads of computer science have pretty clever reasons, talking about powers of two and so on, but avoid finding a simple solution. Therefore, the President has decided to stop arithmetic fatalism and pessimism, and has launched a new movement, called the Strictly Decimal Initiative, aimed at introducing a revolutionary law in mathematics stating that $2^{10} = 1000$. After it was possible to prove that 2^{10} equals 1024, why should a great nation fail to go one step further?

The idea is obviously very attractive, and has been enthusiastically welcomed by many columnists. The idea makes sense, because every child can see that 2^{10} does contain a ten already, so it's logical to have a power of ten as the result. The Numeric Algorithms Tuning Organization, a thoroughly democratic club whose president is by definition a leading U.S. mathematician, immediately said that success of the initiative is only a question of effort. How else could free-world mathematics prove its superiority?

But there are critics, too. Some mathematicians do not even try to hide their sardonic laughter, and insist on the traditional solution. Their usual comment is that while the initiative will not achieve anything, it will certainly absorb some of the best mathematicians for years.

The Soviet Union has not yet reached a standard of computer science that would allow them to compete, but they have warned the U.S. administration that the Strictly Decimal Initiative would certainly threaten the process of mutual binary formula recognition.

European partners are not yet sure whether they should join the U.S. or start their own project, for instance a European programming language that avoids most of the disadvantages of others because it is entirely in French. A spokesman for the U.S. Department of Decimalization has made clear that European cooperation will be appreciated; for safety's sake, however, numbers exceeding 99 will not be passed to anybody outside the U.S.

Though Congress has trimmed the budget proposed by the President, more than peanuts is left. Most mathematicians who have won a contract now take a pragmatic view, trying to solve the problem by stepwise approximation. Some results, obtained on special new hardware, indicate that it should not be beyond our abilities to prove $2^{10} = 1023$. After that, everything else is only a matter of steady financial support.

Jochen Ludewig
ETH Zürich

CALL FOR PAPERS

Call for papers for Computer

Computer magazine seeks articles that cover the state of the art and important new developments in computer science, technology, and applications. Aimed at a broad audience with diverse interests and experience, *Computer* usually publishes surveys or tutorials that facilitate the transfer of technology from university to industry, from research to applications, and across specialized fields. Submit six copies of the manuscript, including illustrations, references, and authors' biographies, to the editor-in-chief:

- Michael C. Mulder
  Applied Research
  University of Portland
  Portland, OR 97203
  Phone (503) 283-7433

Computer also seeks articles on topics relevant to upcoming special issues. Submit six copies of the manuscript directly to the guest editor:

- **FAA's Advanced Automation Program:** Submit by **July 15, 1986,** to G. V. Kloster, 301 Dexter, Denver, CO 80220.

Lastly, *Computer* seeks special issue proposals and articles in the following topic areas:

- microprocessor architectures
- networking and distributed computing
- computer-integrated manufacturing
- software quality assurance
- programming environments

Prospective guest editors and authors should submit proposals and articles directly to Michael Mulder.

An authors' information sheet can be obtained from Michael Mulder at the above address or from the IEEE Computer Society West Coast Office, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720; (714) 821-8380.

IEEE Expert: Articles are solicited for a special issue on building intelligence into the software life cycle. The topic includes AI techniques and tools for developing and maintaining software systems, and software project management. Submitted materials typically (1) range from 15 to 30 double-spaced, typewritten pages (those at the lower end of this page budget are preferred); (2) have 1½-inch margins; and (3) have no more than 10 references. Submit six copies of the article by **June 15, 1986,** to Joseph E. Urban, Center for Advanced Computer Studies, University of Southwestern Louisiana, PO Box 44330, Lafayette, LA 70504; (318) 231-6304.

IEEE Micro: Articles are solicited for a special issue on multiprocessing that is scheduled for publication in October 1986. Submit them by **June 15, 1986,** to James J. Farrell III, VLSI Technology, Inc., 10220 S. 51st St., Phoenix, AZ 85044. Author guidelines may be obtained from James Farrell or from the IEEE Computer Society West Coast Office, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720-2578; (714) 821-8380.

Third International Conference on Data Engineering: February 2-6, 1987, Los Angeles, California. Submit four copies of papers by **June 15, 1986,** to IEEE Computer Society, 1730 Massachusetts Ave., NW, Washington DC 20036-1903; (202) 371-0101.

Compeuro 87, International Conference on VLSI and Computers: May 11-15, 1987, Hamburg, West Germany. Submit summaries (100 words maximum) by **July 1, 1986,** to W. E. Proebster, c/o IBM, PO Box 80 08 80, D-7000 Stuttgart 80, West Germany.

Joint Conference: Melecon 87, Mediterranean Electrotechnical Conference, and 34th Congress on Electronics (IEEE, RIENA): March 24-26, 1987, Rome, Italy. Submit three copies of a 200-word abstract by **June 15, 1986,** to Melecon 87 and 34th Congress on Electronics, Secretariat, c/o RIENA, Via Crescenzio 9, 00193 Rome, Italy. The IEEE Region 8 student paper contest will be held during the conference.

International Switching Symposium (IEEE): March 15-21, 1987, Phoenix, Arizona. Submit six copies of papers (five pages maximum, including a 500-word abstract) by **July 1, 1986,** to Frank Young, Mountain Bell Telephone Co., 3033 N. Third St., Room 901, Phoenix, AZ 85012.

Sixth Annual Phoenix Conference on Computers and Communications (IEEE): February 25-27, 1987, Scottsdale, Arizona. Submit five copies of the paper (5000 words long; include a 200-to-300 word abstract) by **July 3, 1986,** to Forouzan Golshani, Dept. of Computer Science, Arizona State University, Tempe, AZ 85287; (602) 965-2855.

Compcon Spring 87: February 23-26, 1987, San Francisco, California. This conference provides a broad-based technical update on the computer field. Persons interested in making technical presentations should submit four copies of a proposal ranging from 500 words to a draft manuscript for the digest of conference papers. Persons interested in organizing 90-minute sessions involving three speakers should send a description of the proposed session theme and a list of suggested speakers. Submit materials in both categories by **July 11, 1986,** to Glen G. Langdon, Jr., IBM Dept. K54-802, 650 Harry Rd., San Jose, CA 95120-6099; (408) 927-1818.

Fourth International Workshop on Software Specification and Design (ACM, AFCET, Agence de l'Informatique, Alvey Directorate, LCRST-Japan): April 3-4, 1987, Monterey, California. Submit four copies of position papers by **July 14, 1986,** to M. T. Harandi, Dept. of Computer Science, University of Illinois at Urbana-Champaign, 1304 W. Springfield, Urbana, IL 61801.

IEEE Infocom 87: Global Networks—Concept to Realization: March 30-April 2, 1987, San Francisco, California. Submit four copies of the complete paper by **August 1, 1986,** to Izhak Rubin, 6731 Boelter Hall, Electrical Engineering Dept., The University of California at Los Angeles, Los Angeles, CA 90024; (213) 825-2327.

Calls are listed according to submittal deadlines. Conferences that the Computer Society participates in or sponsors are indicated by the IEEE Computer Society logo; others of interest to our readers are also included. For inclusion in "Call for Papers," submit information **six weeks before the month of publication** (e.g., for the **September 1986** issue, send information for receipt by **July 15, 1986**) to COMPUTER, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720.

CALENDAR

June 1986

Forefronts 86, Conference on the Forefronts of Large-Scale Computational Problems (Los Alamos National Laboratory, NSF, SIAM), June 16-18, Troy, New York. Contact Harold J. Raveche, School of Science, Rensselaer Polytechnic Institute, Troy, NY 12180-3590; (518) 266-6305.

International Workshop on Advanced Programming Environments (ACM, IFIP), June 16-18, Trondheim, Norway. Contact Reidar Conradi, Division of Computer Science, Technical University of Norway, N-7034 Trondheim, Norway.

Φ **Symposium on Logic in Computer Science (ACM), June 16-18,** Cambridge, Massachusetts. Contact Ashok K. Chandra, IBM T.J. Watson Research Center, PO Box 218, Yorktown Heights, NY 10598; (914) 945-1752.

Φ **NCC-86, 1986 National Computer Conference (ACM, AFIPS, DPMA, SCS), June 16-19,** Las Vegas, Nevada. Contact AFIPS, 1899 Preston White Dr., Reston, VA 22091; (703) 620-8900.

1986 American Control Conference (AIAA, AIChE, ASME, IEEE, ISA, SCS), June 18-20, Seattle, Washington. Contact Edwin B. Stear, The Washington Technology Center, 376 Loew Hall, FH-10, The University of Washington, Seattle, WA 98195; (206) 545-1920.

1986 AIC Interim Meeting on Color in Computer-Generated Displays (ACM), June 19-20, Toronto, Canada. Contact Peter K. Kaiser, Dept. of Psychology, York University, North York, Ontario M3J 1P3, Canada; phone (416) 667-6335.

ICC-86, International Conference on Communications (IEEE), June 22-25, Toronto, Canada. Contact Hugh J. Swain, Andrew Antenna, Ltd., 606 Beech St., Whitby, Ontario L1N 5S2, Canada; phone (416) 668-3348.

Φ **Computer Vision and Pattern Recognition, June 22-26,** Miami Beach, Florida. Contact Linda Shapiro, MV1, 325 E. Eisenhower, Ann Arbor, MI 48104; (313) 996-8033.

17th Power Modulator Symposium (DoD, IEEE), June 23-25, Seattle, Washington. Contact Bobby Gray, Rome Air Development Center, Griffiss Air Force Base, NY 13441; (315) 330-4846.

CPEM-86, Conference on Precision Electromagnetic Measurement (IEEE), June 23-27, Gaithersburg, Maryland. Contact Norman Belecki, B146 Metrology, National Bureau of Standards, Gaithersburg, MD 20899; (301) 921-2715.

Sigplan 86, Symposium on Compiler Construction (ACM), June 23-27, Palo Alto, California. Contact John R. Sopka, Digital Equipment Corp., ZKO 2-1/P05, 110 Spit Brook Rd., Nashua, NH 03062.

Φ **IEEE Westex 86, June 24-26,** Anaheim, California. Contact Gerhard L. Hollander, Hollander Associates, Box 2276, Fullerton, CA 92633; (714) 879-9005 or Russell Bennett, PO Box 2111, Fullerton, CA 92633-0111; (714) 768-3506.

TI's Second Artificial Intelligence Satellite Symposium: A Step- By-Step Guide to Getting Started, June 25, various sites. Contact Texas Instruments, AI Satellite Symposium II, PO Box 181153, Austin, TX 78715.

Φ **Workshop on Visual Languages, June 25-27,** Dallas, Texas. Contact Robert R. Korfhage, Southern Methodist University, Dept. of Computer Science, School of Engineering and Applied Science, Dallas, TX 75275; (214) 692-3083.

Φ **23rd ACM/IEEE Design Automation Conference, June 29-July 2,** Las Vegas, Nevada. Contact Pat Pistilli, MP Associates, 7366 Old Mill Trail, Suite 101; Boulder, CO 80301; (303) 530-4562.

International Summer Institute: State of the Art in Computer Graphics—Techniques and Applications (ACM, BCS, Computer Graphics Society of Japan), June 29-July 4, Stirling, Scotland. Contact David Rogers, Aerospace Engineering CAD/ICG Group, US Naval Academy, Annapolis, MD 21402 or Frances Johnson, University of Leeds, UK; phone 44-532-459944.

Conferences that the Computer Society participates in or sponsors are indicated by the IEEE Computer Society logo; other conferences of interest to our readers are also included. For inclusion in Calendar, submit information **six weeks before the month of publication** (e.g., for the **September 1986** issue, send information for receipt by **July 15, 1986**) to COMPUTER, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720.

Workshop on Document Generation Principles (ACM), June 30-July 2, Snowbird, Utah. Contact Michael Lesk, Bellcore, Room 2A385, 435 South St., Morristown, NJ 07960; (201) 829-4070.

Fourth IFAC Symposium on Control of Distributed Parameter Systems (AACC, IEEE), June 30-July 3, Pasadena, California. Contact G. Rodriguez, Jet Propulsion Laboratory, Mail Station 198-326, 4800 Oak Grove Dr., Pasadena, CA 91109; (818) 354-4057.

July 1986

Φ **FTCS-16, 16th International Symposium on Fault-Tolerant Computing (IFIP), July 1-3,** Vienna, Austria. Contact H. Kopetz, Institut fur Praktische Informatik, Technische Universitat Wien, Gubhausstraße 30/180, A-1040 Vienna, Austria; phone (0222) 56-01.

International Conference on Systolic Arrays (IEEE), July 2- 4, Oxford, England. Contact Will Moore, Oxford University, Dept. of Engineering Science, Park Rd., Oxford, OX1 3PJ, UK; phone 0865 59988; telex 83295 NUCLOX G.

Second International Conference on Conduction and Breakdown in Solid Dielectrics (IEEE), July 7-10, Erlangen, West Germany. Contact P. Fischer, Siemens AG, Abt. ZFE CWV 2, PO Box 3240, 8520 Erlangen, West Germany; phone 09131-75690.

APL-86 (ACM, BCS), July 7-11, Manchester, England. Contact BISL Conference Dept., The British Computer Society, 13 Mansfield St., London W1M 0BP, UK.

Compass 86, Computer Assurance Conference—Systems Integrity, Process Security and Safety (IEEE), July 7-11, Washington DC. Contact Albert W. Friend, Compass, PO Box 3815, Gaithersburg, MD 20878.

Φ **International Optical Computing Conference, July 7-11,** Shoresh, Israel. Contact Joseph Shamir, Dept. of Electrical Engineering, Technion, Haifa 32000, Israel; phone 04-293273; telex 46406 TECON IL.

Φ **16th Annual Institute in Computer Science, July 7-August 29,** Santa Cruz, California. Contact Ronald C. Smith, University of California Extension, Santa Cruz, CA 95064; (408) 429-2386.

Φ **Workshop on Software Testing (ACM), July 15-17,** Banff, Alberta, Canada. Contact Lori Clark, University of Massachusetts, Amherst, MA 01003; (413) 545-1328.

CAREER OPPORTUNITIES

RATES: $10.00 per line, $100 minimum charge (up to ten lines). Average six typeset words per line, nine lines per column inch. Add $8 for box number. Send copy at least six weeks prior to month of publication to: **Sandra J. Arteaga, Classified Advertising, COMPUTER Magazine, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720.**

In order to conform to the Age Discrimination in Employment Act and to discourage age discrimination, *COMPUTER* may reject any advertisement containing any of these phrases or similar ones: "...recent college grads...," "...1-4 years maximum experience...," "...up to 5 years experience...," or "...10 years maximum experience." *COMPUTER* reserves the right to append to any advertisement, without specific notice to the advertiser, "Experience ranges are suggested minimum requirements, not maximums." *COMPUTER* assumes that, since advertisers have been notified of this policy in advance, they agree that any experience requirements, whether stated as ranges or otherwise, will be construed by the reader as minimum requirements only.

Chairman
Computer Science Department

Concordia University is seeking a chairman for its Computer Science Department. One of the five units in the Faculty of Engineering and Computer Science, the department is one of the largest in Canada and offers undergraduate and graduate programs (to the Ph.D. level) to some 470 full-time and 444 part-time students. There are 26 faculty members. Current research income exceeds $500,000 per year. The major areas of research include: distributed processing and VLSI; computer architecture; software engineering; computer vision; parallel processing; compilers; database management; symbolic computation; operating system; scientific computation.

The Computer Science Department operates a VAX 11/780, running UNIX, and shares a VAX 11/780, running VMS. The Computer Centre operates a VAX 11/750 running VMS, two CDC Cybers 825/830 and 835, and PC networks on both campuses.

The Department is a member of the Centre de Recherche en Informatique de Montréal, a center of excellence for research in Computer Science. This position will interest senior academics with an established track record in teaching and research who are prepared to and capable of assuming a leadership role in the development of computer science in the Faculty, the University and the nation.

Application with curriculum vitae should be sent to:

Dr. M. N. S. Swamy, Dean
Faculty of Engineering and Computer Science
Concordia University
1455 de Maisonneuve Blvd. West
Room H907-7
Montreal, Quebec
H3G 1M8 Canada

In accordance with Canadian immigration requirements, this advertisement is directed to Canadian citizens and permanent residents of Canada, in the first instance.

The University of Alberta
Department of Computing Science

The Department of Computing Science is undergoing an extensive expansion in research initiatives. Applications are invited for three tenure-track positions at the Assistant/Associate Professor level. Responsibilities include research as well as teaching at the graduate and undergraduate levels. Candidates from all areas will be considered. Current hardware support includes an Amdahl 5870, a network of VAX 11/780's, and well equipped mini and micro computer laboratories for graphics, VLSI, and AI research. Access to a Cyber 205 is available. Salary range is $30,316 to $48,970 and is commensurate with qualifications and experience. Send curriculum vitae, names of three references, and up to three reprints or papers. New Ph.D.'s should also include a copy of their transcript. Apply to: Dr. Lee White, Chairman, Department of Computing Science, University of Alberta, Edmonton, Alberta, T6G 2H1. Applications will be accepted until August 31, 1986. The University of Alberta is an equal opportunity employer.

Computer Software Engineer

Formulates mathematical models of systems and sets up and corrects computer systems to solve business and management problems; maintains larger on-line software systems using IBM/370 hardware or similar; designs & develops on-line systems for data center management; consults with originator of problems to determine sources & methods of data collection & methods of determining values of variables; examines and studies verbal descriptions of problems to apply knowledge of scientific discipline and define problems; modifies and implements existing software for other hardware, including Tandem machines; writes, tests, debugs and modifies software using IBM/370 assembler & TSO TEST command; writes program documentation manuals and user guides; assists sales and marketing personnel in demonstrations of these software packages; requires minimum 2 years experience + Bachelor of Science degree in Computer Science or Mathematics, on-line systems design experience, knowledge, and use of TAL, PL/I, and IBM/370 assembler. Job site Los Angeles. Salary $3,000/month. Send this ad and your resume to job #2795, PO Box 865, Sacramento, CA 95894 not later than June 30, 1986.

Systems Analyst

B.S./Computer Science + four years experience or four years Programmer/Analyst. Bi-lingual Japanese/English. Design/Programming experience. NEC's Astra & Proficiency Cobol required. Determine clients' business needs, develop computerized business/inventory/accounting systems for clients. Analyze data, prepare flow charts/diagrams, work with programmers to debug/eliminate system errors, evaluate software/hardware configuration. Prepare instruction manuals. Salary $2683.00 per month, 40 hour week. Location and interview: Los Angeles. Send resume with this ad to Job #MLU1934, P.O. Box 9560, Sacramento, CA 95823-0560, not later than July 1, 1986.

Argonne National Laboratory
Mathematics and Computer Science Division

The Mathematics and Computer Science Division of Argonne National Laboratory is expanding its program in advanced computing research. This program is aimed at developing algorithms, software, and programming techniques for both numerical and symbolic computations on computers with advanced architectures. Emphasis is given to portability across diverse computer architectures. Among the current areas of activity are numerical linear algebra, numerical optimization, numerical integration, partial differential equations, special functions, automated reasoning systems, logic programming, and program transformation systems.

In conjunction with this research program, we have established an Advanced Computing Research Facility (ACRF) that operates a variety of computers with advanced architectures. Currently in operation are an Alliant FX/8 (with vector-parallel capability), an Encore Multimax (with 20 processors sharing 20 megabytes of memory), a Sequent Balance 8000 (with 12 processors sharing 14 megabytes of memory), and an Intel iPSC computer system with a hypercube architecture.

Several job opportunities are available in connection with the ACRF:

- Permanent staff members to conduct research in algorithms and software for computers with advanced architectures. Large-scale scientific computing is an area of particular interest.
- Permanent staff members to manage and operate the ACRF. Applicants should have a strong background in scientific computing and an interest in parallel computing. Responsibilities will include maintaining the ACRF hardware and software, advising ACRF users on how to use the computers in the facility, teaching short courses on parallel computing and ACRF services and software, writing documentation, and providing local enhancements to systems as needed to support our research activities. Knowledge of UNIX® is desirable.
- Short-term and long-term visitors with research interests in algorithms and software for computers with advanced architectures. Highest priority will be given to applicants whose research interests are in areas that could benefit our existing research projects and who are already active in parallel computing research.

For more technical information, write Jack Dongarra or Rusty Lusk, Advanced Computing Research Facility, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois 60439.

Resumes should be forwarded to:

Rosalie L. Bottino
Box J-MCS-2K
ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, IL 60439
Argonne is an equal opportunity/
affirmative action employer.

ATE Consultants

Needed by major ATE consulting firm. Long and short term assignments throughout U.S. Experienced only. Send resume to: Box CLM, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720.

University of Auckland
New Zealand
Chair In Computer Science

The Department of Computer Science at The University of Auckland invites applications for a newly established second Chair.

Applicants should be qualified Computer Scientists whose personal qualities and experience will enable them to contribute significantly to the continuing development of Computer Science at Auckland both in teaching and research. As the successful candidate will be expected to play a leading role in the promotion of research within the department, a demonstrated history of research administration and liaison with industry and external agencies is expected in addition to an accomplished research record.

Applicants should be prepared to share the departmental administrative load including, eventually, periods as Head of the Department.

Commencing salary will be established in the range NZ$62,000-77,500 per annum, having regard for qualifications and experience of the candidate concerned.

Further information including Conditions of Appointment and Method of Application are available from the Assistant Registrar (Academic Appointments), University of Auckland. Applications should be forwarded before the closing date of 30 June 1986.

W. B. Nicoll, REGISTRAR, University of Auckland, Private Bag, Auckland, NEW ZEALAND.

Michigan State University

The Department of Computer Science invites applications for tenure track positions at all levels. Candidates from all areas of specialization in computer science or computer engineering will be considered. The department has a special interest in candidates in the areas of programming languages, database systems, artificial intelligence and expert systems, robotics, design of computer systems and networks, parallel computation, dataflow machines, operating systems and computational complexity. Candidates should have a Ph.D. in computer science or computer engineering and have a strong interest in both research and teaching. Applications will be accepted until the positions are filled.

As a unit within the College of Engineering at Michigan State University, Computer Science offers the Bachelor of Science, Master of Science and Doctor of Philosophy degrees. Special support is available from within the college and university to initiate research by new faculty members. Michigan State University enjoys a park-like campus of 2,100 developed acres and 3,100 acres of experimental farms, outlying research facilities and natural areas. The campus is adjacent to the cities of East Lansing and the capital city, Lansing. The Greater Lansing area has approximately 250,000 residents. The communities have fine school systems and place a high value on education.

Applicants should send a resume and a statement of research and teaching interests to:

Dr. Anthony S. Wojcik, Chairperson
Department of Computer Science
A714 Wells Hall
Michigan State University
East Lansing, Michigan 48824-1027
CSNET: wojcik@mich-state

Michigan State University is an Equal Opportunity/Affirmative Action Institution and encourages applications from members of ethnic minority groups.

University of California, Berkeley

Associate Research Engineer, Electronics Research Laboratory, University of California, Berkeley. Temporary full time position for one year commencing August 1986, with possibility of extension contingent upon funding. Salary $51,400 to $55,500 per annum, dependent on qualifications. Assist in computer architecture research with professor and graduate students. Develop architectures for high performance Prolog machines and implement the architectures wiring CMOS-VLSI technology. Ph.D. degree or equivalent in Computer Science or related field required, and research experience in computer architecture, VLSI design, testing of digital systems, use and design of silicon compilers for special purpose architectures. Applicants must also have experience with analysis and design of high performance computer systems. Send resume to Professor A. Despain by June 30, 1986, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720. The University of California is an equal opportunity, affirmative action employer.

Portland State University
Position Announcement
In Electrical Engineering

The Department of Electrical Engineering has tenure-track faculty positions at the Assistant, Associate or Full Professor level available immediately. Applicants must have an earned doctorate. Areas of particular interest are *VLSI Design, Interactive Computer Graphics, Computer Architecture,* and *Operating Systems.* A VLSI design center, computer vision laboratory, and optical communications laboratory have recently been established in the Portland Center for Advanced Technology. Faculty members have access to a number of computer installations for research purposes including a *VAX 11/780, IBM 4381, Gould 9000,* a *number of PDP 11 installations* and a *CDC Cyber system.* In addition, the Department of Electrical Engineering has state-of-the-art laboratory facilities in computer graphics and microcomputer systems. Responsibilities include undergraduate and graduate teaching, development of sponsored research and interaction with local industry.

Portland has a rapidly-growing computer and electronics industry including *Tektronix (world headquarters), Intel, Hewlett-Packard, Floating Point Systems (headquarters), Electro-Scientific Industries, Northwest Instruments, Sequent Computer Systems, Metheus, Mentor Graphics, Oregon Software, Lattice, etc.,* which permits close industry-university interaction.

During the past three years, *the computer and electronics industry, the State of Oregon, and the City of Portland* have launched major initiatives to expand computer science and electrical engineering education and research. These include *$3 million distributed by the Oregon High Technology Consortium, a $3.5 million five-year program funded by the Tektronix Foundation, and the* acquisition of a *30,000 square foot facility (the Portland Center for Advanced Technology),* which houses the Departments of Computer Science and Electrical Engineering.

While this search is primarily for experienced faculty, we also strongly encourage recent graduates to apply. Rank and salary are commensurate with qualifications and experience. Send an application, including a resume listing the names of three references to:

Dr. P. A. Frick, Department Head
Electrical Engineering Department
Portland State University
P.O. Box 751
Portland, Oregon 97207
Telephone: (503) 229-3806

Non-U.S. residents must state their visa status. Portland State University is an equal opportunity/affirmative action employer. Qualified minorities, women, and members of other protected groups are encouraged to apply.

COMPUTER SCIENTIST

Cranston/Csuri Productions is currently searching for qualified applicants for one or more positions of Computer Graphics Software Specialists. Of particular interest are individuals with experience in the areas of display algorithms, geometric modeling, and systems software. C/CP has a staff of 40 professionals, including animation and technical personnel in addition to computer science personnel. C/CP utilizes DEC VAX, Pyramid, and SUN computer systems, with internally designed frame buffers outputting to video or film recorders.The software development environment consists of Unix with C. Interested parties should submit a resume, transcripts, and three letters of reference to: Doreen P. Close, Computer Graphics Software Specialist, Cranston/ Csuri Productions, Inc., 1501 Neil Ave., Columbus, OH 43201. Preference will be given to those individuals with a Master of Science degree in Computer Science.

College of Science
Department of Computer Sciences
Al-Jadyria, Baghdad, Iraq

The Computer Science Department of the University of Baghdad, College of Science invites applications for faculty positions at all ranks for one year or more. Applicants should have Ph.D. in computer science or computer engineering in one of the following areas: operating systems, compiler constructions, file organization and database, computer network, software engineering, information systems.

The department offers both B.S. and M.S. degrees. The computer facilities include two Unix-based systems, many micros, microprocessor-development lab and digital logic lab.

The applicants should have commitments for both teaching and research. Applications with available certificates should be mailed to the following address:

Chairman
Department of Computer Science
College of Science
University of Baghdad
Al-Jadyria, Baghdad
Iraq Telex 212197

Dahlgren Chair
Computer Science
Virginia Tech

Applications and nominations of distinguished computer scientists are sought for the John Adolphus Dahlgren Chair, created in the Department of Computer Science at Virginia Tech through a research agreement with the U.S. Navy. The Chair provides unique opportunities for large-scale research projects through the university's multidisciplinary Systems Research Center.

Candidates with distinguished records in research and scholarship are sought who can exercise a leadership role among the 20 computer science faculty and also provide guidance to research programs pursued by Naval laboratories. Candidates are particularly sought who have expertise in software engineering, artificial intelligence, distributed processing systems, human/computer interaction, and/or performance analysis.

To submit applications or nominations, or for additional information, contact Dr. Dennis Kafura, Dahlgren Chair Search Committee, Department of Computer Science, Virginia Tech, Blacksburg, VA 24061 by mail; (703) 961-6931 by phone; or "kafura@vpi" by CSNET. The search will continue until the position is filled.

Virginia Tech Is an Equal Opportunity/Affirmative Action Employer.

ADVERTISER INDEX

ACM 25
AFIPS 28-29
BDM .... Cover II
Boeing .... Cover III
CACI 1
COMPEURO '87 24
Co-efficient Precision 41
FJCC 18
IEEE Computer Society Membership 30-31
IEEE Computer Society Press 40
IPS IEEE/PES 5
Jackson Chamber of Commerce 27
NCR Corp 17
Office of Naval Research 77
Presearch 4
Springer-Verlag 56
Texas Instruments 8-9
Tutorial Week Program 16
United Technologies Research Center 71
University at Buffalo 4
Watcom Products .... Cover III
Classified Advertising 94-95

FOR DISPLAY ADVERTISING INFORMATION CONTACT

Southern California and Mountain States: Richard C. Faust Company, 24050 Madison Street, Suite 100, Torrance, CA 90505; (213) 373-9604.
Northern California and Pacific Northwest: Don Farris Company, 161 W. 25th Ave., #102B, San Mateo, CA 94403; (415) 349-2222.
Jack Vance, P.O. Box 3205, Saratoga, CA 95070; (408) 741-0354.
East Coast: Hart Associates, Inc., P.O. Box 339, 42 Lake Blvd., Matawan, NJ 07747; (201) 583-8500.
New England: Arpin Associates, P.O. Box 227, Weston, MA 02193; (617) 899-5613.
George Watts, III, 4 Conifer Dr., Wilbraham, MA 01095; (413) 596-4747.
Midwest: Thomas Knorr, Knorr MicroMedia, Inc., 333 North Michigan Ave., Chicago, IL 60601; (312) 726-2633.
Southeast: Larry C. Shattles, 133 Laurel Oak Drive, Longwood, FL 32779; (305) 788-1950.
Southwest: The House Company, 3817 Richmond Avenue, Suite 110, Houston, TX 77027; (713) 622-2868.

Advertising Manager: Mike Koehler, *COMPUTER* Magazine, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720; (714) 821-3240, 821-8380.

For production information, conference or classified advertising contact Sandra J. Arteaga, or Carole Porter, *COMPUTER* Magazine, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720; (714) 821-1140.

PRODUCT INDEX

	RS#	Page#
A.L.P. Systems	47	83
Analogic Corp.	80	86
CMS	81	86
Co-efficient Precision	2	41
Communication Machinery Corp.	82	86
Dual Systems Corp.	43	82
Electronic Vaults Inc.	83	86
Excelan Inc.	84	86
Flexible Computer Corp.	45	82
Floating Point Systems, Inc.	41	81
Honeywell	70	88
IBM	46	83
Ideassociates, Inc.	85	86
Imaging Technology Inc.	86	86
IMSL	50	84
Intech, Inc.	71	88
Intel Corp.	51	84
Interaction Systems, Inc.	44	82
Jackson Chamber of Commerce	1	27
Lifeboat Associates	87	86
Light Machines Corp.	88	86
Lisp Machine, Inc.	52	84
Matrox Electronic Systems Ltd.	89	87
MDB Systems, Inc.	48	83
Metheus Corp.	55	85
National Semiconductor Corp.	72	88
Oki Semiconductor Corp.	73	88
Oregon Micro Systems Inc.	90	87
Plessey Microsystems	91	87
Precision Monolithics Inc.	74	88
Puma Software, Inc.	54	85
Quadram	92	87
Rapid Systems, Inc.	93	87
Signetics Corp.	75	88
Silicon Systems	76	88
Spencer Organization, Inc.	94	87
Springer-Verlag	3	56
SPS Software Products & Services, Inc.	95	87
Standard Microsystems Corp.	77,96	88,87
Star Gate Technologies	97	87
Techware	98	87
Texas Instruments Inc.	42	82
Thinking Machines Corp.	40	81
Toshiba America, Inc.	78	88
Unimation Inc.	53	85
Watcom Products	4	Cover IV
Weitek Corp.	79	88
Wendin	99	87
ZyLab Corp.	49	84

COMPUTER READER SERVICE

For further information on advertised products, new products, or literature and to specify levels of interest in *Computer*'s articles and departments fill out one of these cards. Affix stamp and mail.

Please print or type your name and address.

Reader Service Card

COMPUTER

Void after 12/15/86

6/86

Name ______________________

Company ______________________

Title ______________________

Address ______________________

City ______________ State ______________

Zip ______________ Telephone ______________

Circle your level of interest in the following articles and departments:

	High	Medium	Low	Did not read
Guest Editors' Intro	101	102	103	104
America's Future	105	106	107	108
Design Education	109	110	111	112
Low Cost Workstations	113	114	115	116
Vocationalism	117	118	119	120
Accreditation	121	122	123	124
CS in China	125	126	127	128
Japanese Software	129	130	131	132
Visual Browsing	133	134	135	136
Workstation	137	138	139	140
Update	141	142	143	144
Open Channel	145	146	147	148

I need more information on the following:

1	2	3	4	5	6	7	8	9	10	11	12	13	14
15	16	17	18	19	20	21	22	23	24	25	26	27	28
29	30	31	32	33	34	35	36	37	38	39	40	41	42
43	44	45	46	47	48	49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80	81	82	83	84
85	86	87	88	89	90	91	92	93	94	95	96	97	98

Products purchased or specified:

	Hobby	Job
Computers	150	151
Peripherals	152	153
Data Communication	154	155
Memories/components	156	157
Software and Services	158	159
Publications	160	161

170 Please send me the IEEE-CS 1986 Publications catalog.
171 Please send me an application for advancement to senior member.
172 Please send me an IEEE-CS membership application.

Reader Service Card

COMPUTER

Void after 12/15/86

6/86

Name ______________________

Company ______________________

Title ______________________

Address ______________________

City ______________ State ______________

Zip ______________ Telephone ______________

Circle your level of interest in the following articles and departments:

	High	Medium	Low	Did not read
Guest Editors' Intro	101	102	103	104
America's Future	105	106	107	108
Design Education	109	110	111	112
Low Cost Workstations	113	114	115	116
Vocationalism	117	118	119	120
Accreditation	121	122	123	124
CS in China	125	126	127	128
Japanese Software	129	130	131	132
Visual Browsing	133	134	135	136
Workstation	137	138	139	140
Update	141	142	143	144
Open Channel	145	146	147	148

I need more information on the following:

1	2	3	4	5	6	7	8	9	10	11	12	13	14
15	16	17	18	19	20	21	22	23	24	25	26	27	28
29	30	31	32	33	34	35	36	37	38	39	40	41	42
43	44	45	46	47	48	49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80	81	82	83	84
85	86	87	88	89	90	91	92	93	94	95	96	97	98

Products purchased or specified:

	Hobby	Job
Computers	150	151
Peripherals	152	153
Data Communication	154	155
Memories/components	156	157
Software and Services	158	159
Publications	160	161

170 Please send me the IEEE-CS 1986 Publications catalog.
171 Please send me an application for advancement to senior member.
172 Please send me an IEEE-CS membership application.

Comments on articles and other editorial matter

What did you find appealing about the articles/departments you liked?

What other topics/departments would you like to see?

General comments:

For reader service, see other side

This PO box far reader
service cords only.

PLACE
STAMP
HERE

COMPUTER
magazine

Reader Service Inquires
Box 24008
Los Angeles, Ca 90024
USA

Comments on articles and other editorial matter

What did you find appealing about the articles/departments you liked?

What other topics/departments would you like to see?

General comments:

For reader service, see other side

This PO box for reoder
service cards only.

PLACE
STAMP
HERE

COMPUTER
magazine

Reader Service Inquires
Box 24008
Los Angeles, Ca 90024
USA

Open new doors at Boeing

Some of the most respected scientific minds in the country are opening new doors to significant advancements in electronics.

If you're qualified, you could join them at the new Boeing Electronics High Technology Center in Bellevue, Washington. Just minutes from the University of Washington and Seattle. In an attractive, creative environment for advanced research, away from distractions.

The Center is equipped with state of the art laboratories where you can develop, test and perfect your ideas.

The positions are open to candidates with advanced degrees (MS or Ph. D) in Electronic Engineering, Physics, Physical Chemistry, Materials Science, Computer Science or Applied Mathematics.

If you believe as we do, that the most exciting era in Electronics, Electronics/Photonics has just begun, send your résumé, with present and expected salary, to The Boeing Company, P.O. Box 3707-DBD, Seattle, WA 98124. Who knows. You may be the next person to cross the threshold to an electronics/photonics breakthrough.

BOEING

Boeing needs world-class engineers and scientists for these specialties

Photonics

Fiber optics sensors

Wideband data devices and components

Infrared and visible spectrum sensors

Optical communications

Optical information processing

Radio Frequency

Monolithic microwave integrated circuits

Millimeter wave integrated circuit technology

Advanced devices and circuit components

Secure, reliable transmitters and receivers

Microelectronics

Radiation hardened circuits and devices

Advanced group III-V circuit technology

Advanced microelectronics packaging and interconnects

Advanced integrated circuits

Information Processing

Ultrareliable computer architectures

High performance computer architectures

Signal and image processing

Symbolic processing

Machine vision

Advanced display concepts

Materials Processing

Crystal growth (III-V, II-VI)

Bulk and thin film device fabrication

Materials and device characterization

Independent Research

Advanced electro-optic materials

3D structures

Innovative concepts

Get the Most Out of Your Courses with WATCOM Programming Languages

Programmer Efficiency
WATCOM Programming Languages have been developed to optimize programmer efficiency. With features such as integrated debugging, accurate and informative error diagnostics, and one-step compilation and execution, WATCOM's languages are perfect for special academic environments. Students accomplish more without technical consultation. Time spent debugging programs is minimized.

Academic Expertise
The University of Waterloo has been developing academic languages since 1965. In affiliation with the University, WATCOM distributes the wide range of languages from both the University of Waterloo and WATCOM. This creative partnership together brings together the expertise, experience and support needed to offer programming languages that match the unique requirements of education.

Language Selection
WATCOM has an outstanding collection of languages available for IBM mainframes, IBM PC's,* and Digital VAX's.* Load-and-go compilers include WATFOR-77* (FORTRAN), WATBOL* (COBOL), and Waterloo PASCAL. Waterloo BASIC is our incremental compiler. And WATCOM Interpreters include APL, BASIC, COBOL, and Pascal.

Support
Precise documentation is available with WATCOM's comprehensive textbooks, manuals, primers, and tutorials. Our regular software product releases and Newsletters keep you up to date. And, WATCOM's telephone Hotline is always there when you need it.

WATCOM – more than just software

With more than 200,000 licenses worldwide, WATCOM is a leading developer of educational software tools. WATCOM means compilers, interpreters, and local area networks. Plus document composition and data management applications. All WATCOM products are engineered for optimum human efficiency and productivity.
But we're more than just software. Our people's active involvement with educators spans two decades, enabling us to produce and support software which meets specific academic requirements. WATCOM is a corporate partner of Waterloo's Institute for Computer Research and of EDUCOM. Partnerships such as these keep WATCOM on the leading edge of software developments and academic requirements.
Make the right choice. Call or write WATCOM today and we'll tell you all about our people-efficient products.

Yes! I want to make the right choice in software. IEE-11
Send me more information on:
☐ WATCOM Programming Languages
☐ WATCOM Software Catalogue.

Name: ________________

Company: ________________

Title: ________________

Address: ________________

City: ________________

State: __________ Zip: ______

WATCOM
The right choice in software.

WATCOM PRODUCTS INC.
415 Phillip Street
Waterloo, Ontario, Canada
N2L 3X2
(519) 886-3700
Telex 06-955458

*WATBOL and WATFOR are registered trademarks of the University of Waterloo.
*IBM and IBM PC are registered trademarks of International Business Machines Corporation.
*VAX is a registered trademark of Digital Equipment Corporation.

Reader Service Number 4